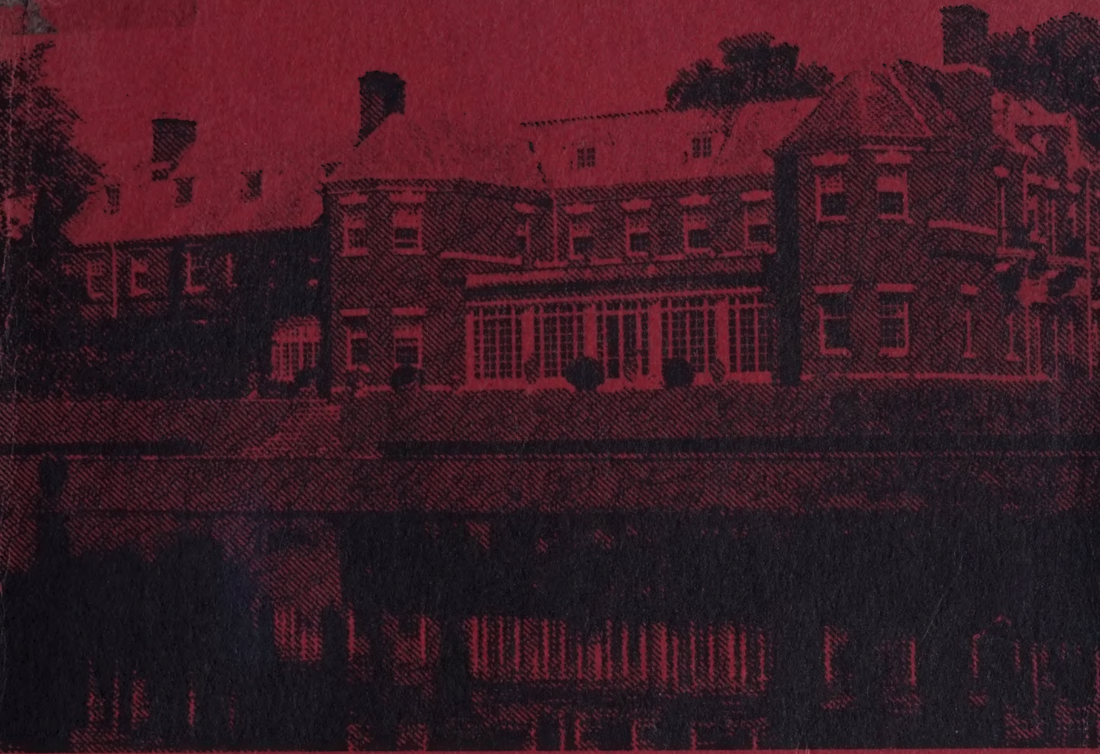
Engin.



Proceedings

TWENTIETH ANNUAL ALLERTON CONFERENCE ON COMMUNICATION, CONTROL, AND COMPUTING

October 6-8, 1982

Allerton House, Monticello, Illinois
Sponsored by the
Department of Electrical Engineering and the
Coordinated Science Laboratory of the
University of Illinois at Urbana-Champaign

The Library of the
MAY 1984
University of Illinois

The person charging this material is responsible for its return to the library from which it was withdrawn on or before the **Latest Date** stamped below.

Theft, mutilation, and underlining of books are reasons for disciplinary action and may result in dismissal from the University.
To renew call Telephone Center, 333-8400

UNIVERSITY OF ILLINOIS LIBRARY AT URBANA-CHAMPAIGN

ENGINEERING

SEP 23 1984
JAN 13 REC'D
PHOTO REPRODUCTION
PHOTO REPRODUCTION
JUN 19 REC'D
SEP 2 REC'D
PHOTO REPRODUCTION
SEP 20 REC'D
JUL 15 REC'D
OCT 21 1985
PHOTO REPRODUCTION
OCT 14 REC'D
OCT 07 REC'D
INTERLIBRARY LOAN
APR 15 1993
NOV 18 REC'D
JUN 03 REC'D
INTERLIBRARY LOAN
JUL 02 1993
MAY 15 REC'D
MAR 22 1987
MAR 06 REC'D

L161—O-1096

PROCEEDINGS

TWENTIETH ANNUAL ALLERTON CONFERENCE ON COMMUNICATION, CONTROL, AND COMPUTING

H. V. Poor
W. K. Jenkins
Conference Co-Chairmen

Conference held
October 6-8, 1982
Allerton House
Monticello, Illinois

Sponsored by
The Department of Electrical Engineering
and
The Coordinated Science Laboratory
of the
UNIVERSTIY OF ILLINOIS
at
Urbana-Champaign

Engineering

FOREWORD

The Twentieth Annual Allerton Conference on Communication, Control and Computing was held at Allerton House, Monticello, Illinois on October 6, 7, and 8, 1982. The Conference was sponsored by the Department of Electrical Engineering and the Coordinated Science Laboratory of the University of Illinois at Urbana-Champaign. These PROCEEDINGS contain the papers presented at the Conference.

In addition to the large number of contributed papers, there were several invited papers. We are grateful to Mr. William F. Powers of Ford Motor Company for organizing and chairing a session of invited papers on Flexible Manufacturing Systems. His time and efforts on behalf of the Conference are deeply appreciated. We would also like to thank Professors Robert J. McEliece of the California Institute of Technology and Dharma P. Agrawal of North Carolina State University for accepting our invitation to participate in the session on Coding, and to express our gratitude to the other session chairmen and authors for their contributions to the success of the Conference. The musical contributions of Professor Daniel J. Perrino and his band, Medicare 7, 8, or 9, to the social program are also gratefully acknowledged.

On Wednesday afternoon Professor Mac E. Van Valkenburg presented his perspectives on the Allerton Conference in a plenary address entitled "The First Twenty Years of the Allerton Conferences." We are very grateful to him for his presentation and to Professor Jose B. Cruz, Jr. for serving as chairman of this plenary session.

The support and encouragement of Professor G. W. Swenson, Jr., Head of the Department of Electrical Engineering, and Professor R. T. Chien, Director of the Coordinated Science Laboratory are appreciated and acknowledged. We would also like to thank our colleagues, Professors J. A. Abraham, T. Basar, T. Ü. Basar, D. J. Brown, J. B. Cruz, Jr., B. E. Hajek, I. N. Hajj, T. S. Huang, P. V. Kokotovic, D. P. Looze, M. C. Loui, D. E. Muller, D. C. Munson, Jr., J. H. Patel, W. R. Perkins, F. P. Preparata, M. B. Pursley, D. V. Sarwate, T. N. Trick, and M. E. Van Valkenburg for their generous assistance in organizing the Conference.

We are grateful to Mr. Steven Williams for his handling of the business affairs of the Conference and to Ms. Rose Harris for her assistance with Conference registration. Finally, very special thanks are due to Mrs. Rosemary Wegeng and Mrs. Phyllis Young for their gracious and impeccable assistance in all aspects of the organization of the Conference, from the initial preparation of the Conference announcement through Conference registration to the final editing of these PROCEEDINGS.

H. V. Poor

W. K. Jenkins

TABLE OF CONTENTS

FOREWORD.. iii

I-A: COMMUNICATION NETWORKS

Chairman: A. E. Ephremides
University of Maryland

TRANSIENT ANALYSIS OF A PACKET SWITCHING NODE WITH A BUFFER ADMISSION SCHEME........................ 1
G. B. Agnew and J. W. Mark

DATA LINK FLOW CONTROL WITH PARTIAL OBSERVATIONS....... 3
A. A. Lazar

ON A NEW LOCAL AREA NETWORK HYBRID ACCESSING SCHEME APPLYING SPLIT-CHANNEL ALLOCATION.................... 5
B. Dimitriadis, N. Alexandridis and G. Polyzos

DYNAMIC OPTIMAL CONTROL OF A M/M/1 QUEUE.............. 15
F. Vakil and A. A. Lazar

COMPUTING RELIABILITY FOR A GENERALIZATION OF SERIES-PARALLEL NETWORKS.............................. 25
J. A. Wald and C. J. Colbourn

AN AVERAGE MESSAGE DELAY CONSTRAINED MINIMUM HOP FLOW ASSIGNMENT AND ROUTING ALGORITHM.................. 27
J. S. Meditch and M. S. Chen

I-B: COMPUTATIONAL GEOMETRY

Chairman: F. P. Preparata
University of Illinois

THE ULTIMATE PLANAR CONVEX HULL ALGORITHM?............ 35
D. G. Kirkpatrick and R. Seidel

ELIMINATING HIDDEN-LINES FROM MONOTONE SLABS.......... 43
D. Rappaport

MINIMUM EDGE LENGTH PARTITIONING OF RECTILINEAR POLYGONS................................... 53
A. Lingas, R. Y. Pinter, R. L. Rivest, A. Shamir

AN O(nlogn) ALGORITHM FOR DECOMPOSING SIMPLE RECTILINEAR POLYGONS INTO CONVEX QUADRILATERALS........ 64
J.-R. Sack

THE COMPLEXITY OF COMPUTING MINIMUM CONVEX COVERS FOR POLYGONS................................. 75
J. O'Rourke

AN OPTIMAL ALGORITHM TO CONSTRUCT ALL VORONOI DIAGRAMS FOR K NEAREST NEIGHBOR SEARCHING IN THE EUCLIDEAN PLANE.................................... 85
F. Dehne

THE COMPLEXITY OF VORONOI DIAGRAMS IN HIGHER DIMENSIONS.................................... 94
R. Seidel

I-C: IMAGE AND SIGNAL PROCESSING

Chairman: D. C. Munson, Jr.
University of Illinois

A NOTE ON PRECISION AND ILL-POSEDNESS IN LINEAR IMAGING PROBLEMS.................................... 97
L. S. Joyce and W. L. Root

A SIMPLE NONLINEAR EDGE PRESERVING FILTER.............. 107
C. A. Pomalaza and C. D. McGillem

EDGE DETECTION USING CONTOUR TRACING.................. 116
E. J. Delp and C.-H. H. Chu

AN ALGEBRAIC DESCRIPTION OF ARRAY IMPLEMENTATIONS OF FFT ALGORITHMS...................... 126
L. Johnsson and D. Cohen

ESTIMATION OF MOTION PARAMETERS FROM LOCAL MEASUREMENTS OF VELOCITY COMPONENTS................... 135
E. J. Delp and P. H. Eichel

FREQUENCY DOMAIN ALIGNMENT OF DISCRETE-TIME SIGNALS.... 144
M. Simaan

A VLSI IMPLEMENTATION OF MEMORY INTENSIVE RESIDUE NUMBER SYSTEM ARCHITECTURE.................... 150
M. A. Bayoumi, G. A. Jullien and M. A. Sid-Ahmed

I-D: LINEAR SYSTEMS

Chairman: S. Bingulac
University of Belgrade, Yugoslavia

INTERNAL ZEROS AND THE SYSTEM MATRIX.................. 153
B. F. Wyman and M. K. Sain

AN EFFICIENT POLE PLACEMENT BY OUTPUT FEEDBACK......... 159
S. Bingulac and J. Medanić

TOWARDS A FUZZY GEOMETRIC THEORY OF LINEAR MULTIVARIABLE CONTROL.......................... 167
H. S. Ramirez

CONTROL CONCEPTS FOR FINITE STATE SYSTEMS AND INVERTIBILITY CRITERION...................... 177
T. Nomura

RICCATI EQUATIONS AND THE MATRIX EIGENVALUE PROBLEM..... 187
G. Ammar and C. Martin

STRUCTURAL STABILIZATION OF UNCERTAIN SYSTEMS: NECESSITY OF THE MATCHING CONDITION..................... 196
I. R. Petersen

I-E: TESTING AND DIAGNOSIS

Chairman: C. P. Kruskal
University of Illinois

A FILTERING APPROACH TO TRANSIENT FAULT DIAGNOSIS....... 206
M. A. Kennedy and G. G. L. Meyer

FAULT LOCATION IN SEQUENTIAL TREES..................... 209
S. Chandra and K. K. Saluja

ON THE COMPLEXITY OF ESTIMATING THE SIZE OF A TEST SET.................................. 219
B. Krishnamurthy and S. B. Akers

GENERALIZED DATA COMPRESSION FOR VLSI TESTING.......... 229
K. K. Saluja and C. S. Venkatraman

ADAPTIVE DIAGNOSIS: A NEW THEORY OF t-FAULT-DIAGNOSABLE SYSTEMS......................... 231
S. L. Hakimi and K. Nakajima

ON A DIAGNOSTIC CONFIDENCE MEASURE FOR A CLASS OF DIGITAL SYSTEMS............................ 241
I. S. Hwang and C. R. Kime

II-A: PARALLEL AND DISTRIBUTED COMPUTATION

Chairman: D. E. Muller
University of Illinois

ORDER STATISTICS ON DISTRIBUTED SETS.................... 251
N. Santoro and J. B. Sidney

PARALLEL GRAPH ALGORITHMS WITHOUT MEMORY CONFLICTS...... 257
D. S. Hirschberg

THE M-SHUFFLE AS AN INTERCONNECTION NETWORK FOR SIMD MACHINES.. 264
K. Keutzer and E. Robertson

PARSING DYCK LANGUAGES ON PARALLEL MACHINES............ 272
R. M. Mattheyses and C. M. Fiduccia

II-B: QUANTIZATION

Chairman: S. A. Kassam
University of Pennsylvania

QUANTIZATION WITH NON-DIFFERENCE BASED DISTORTION MEASURES.................................. 281
E. F. Abaya and G. L. Wise

OPTIMUM QUANTIZER PERFORMANCE FOR A CLASS OF NON-GAUSSIAN MEMORYLESS SOURCES...................... 290
N. Farvardin and J. W. Modestino

AN ALGORITHM FOR DESIGNING VECTOR QUANTIZERS............ 301
K. Sayood and J. D. Gibson

SPHERICAL COORDINATES QUANTIZERS: ASYMPTOTIC ANALYSIS AND THE THREE-DIMENSIONAL CASE...... 311
P. F. Swaszek

II-C: IDENTIFICATION AND NONLINEAR CONTROL

Chairman: P. J. Antsaklis
University of Notre Dame

IDENTIFICATION OF CONSERVATIVE LINEAR MECHANICAL SYSTEMS..................................... 321
S. W. Sirlin, R. W. Longman and J. N. Juang

A METHOD FOR SIMULTANEOUS ESTIMATION OF NOISE COVARIANCES AND PLANT IDENTIFICATION................... 331
Y. G. Zhang and R. W. Longman

A BILINEAR OBSERVER APPROACH TO A CLASS OF NON-LINEAR STATE RECONSTRUCTION PROBLEMS................ 339
H. S. Ramirez

OBSERVER METHODS IN SELF-TUNING CONTROL................. 341
K. Warwick and P. J. Antsaklis

A DIGITAL COMPENSATION TECHNIQUE FOR PRESSURE TRANSDUCERS.................................... 343
C. E. Lindahl

II-D: RANDOM PROCESSES

Chairman: S. C. Schwartz
Princeton University

STOCHASTIC RELIABILITY FUNCTIONS FOR FAILURE RATES DERIVED FROM GAUSS-MARKOV PROCESSES AND FILTERS WITH POINT PROCESS OBSERVATIONS........................ 345
J. L. Hibey

CHARACTERIZATION AND ESTIMATION
OF TWO DIMENSIONAL ARMA MODELS.......................... 355
R. L. Kashyap

STOCHASTIC CHARACTERIZATION OF OSCILLATOR
PHASE INSTABILITIES..................................... 363
V. C. Vanicola and P. K. Varshney

EFFECTS OF A ZERO MEMORY NONLINEARITY ON THE
SPECTRUM OF A RANDOM PROCESS............................ 366
G. L. Wise

II-E: FOUNDATIONS OF COMPUTING

Chairman: N. Dershowitz
University of Illinois

TIME-SPACE TRADEOFFS FOR BACK-TO-BACK FFTs:
THE EFFECT OF THE CONNECTION PATTERN..................... 374
D. A. Carlson

CONTEXT-FREE INTERSECTION OF
CONTEXT-FREE LANGUAGES.................................. 384
S. Kundu

FINITE AUTOMATA WITH SEVERAL INITIAL STATES............. 394
P. A. S. Veloso

INCREMENTAL UNIFICATION IN EQUATIONAL THEORIES.......... 396
J. P. Jouannaud, C. Kirchner and H. Kirchner

CHARACTERIZATION OF WEIGHT VECTORS FOR
WEIGHTED SYNCHRONIC DISTANCE IN PETRI NETS............... 406
I. Suzuki and T. Kasami

III-A: FLEXIBLE MANUFACTURING SYSTEMS

Organizer and Chairman: W. F. Powers
Ford Motor Company

LARGE SCALE NONLINEAR PROGRAMMING....................... 416
L. S. Lasdon and A. D. Waren

✓ A HIERARCHICAL APPROACH TO PRODUCTION PLANNING
IN FLEXIBLE MANUFACTURING SYSTEMS....................... 426
K. E. Stecke

✓ OPTIMIZATION OF FLEXIBLE MANUFACTURING SYSTEMS
USING NEW TECHNIQUES IN DISCRETE EVENT SYSTEMS.......... 434
R. Suri and X. Cao

/ MODELING A MULTI-MACHINE WORK STATION IN
FLEXIBLE MANUFACTURING SYSTEMS (FMS).................... 444
D. D. W. Yao and J. A. Buzacott

III-B: DATA STRUCTURES

Chairman: C. L. Liu
University of Illinois

WORST-CASE ANALYSES OF SELF-ORGANIZING SEQUENTIAL SEARCH HEURISTICS.............................. 452
J. L. Bentley

A NEW TIME COMPLEXITY FOR ORTHOGONAL RANGE QUERIES.. 462
D. E. Willard

SOME PROGRAMMING CONCEPTS FOR CONCURRENT DELETION IN B-TREES.................................... 472
Y. S. Kwong and D. Wood

BINARY SEARCH TREES WITH LOCAL ROTATION................ 481
S.-H. S. Huang and C. K. Wong

RECURSIVELY ROTATED ORDERS AND IMPLICIT DATA STRUCTURES: A LOWER BOUND.......................... 490
G. N. Frederickson

PROBABILISTIC SEARCHING IN SORTED LINKED LISTS.......... 500
T. Leighton and M. Lepley

III-C: ESTIMATION

Chairman: J. B. Thomas
Princeton University

DYNAMIC PROGRAMMING, LEVINSON ALGORITHM, AND LATTICE EQUALIZERS.................................. 507
F. Pollara and K. Yao

MINIMAX LOCATION PARAMETER ESTIMATION UNDER UNIFORM LOSS WITH A COMPACT PARAMETER SPACE............ 516
M. Zeytinoglu and M. Mintz

SUFFICIENT STATISTICS FOR STATIONARY GAUSSIAN RANDOM PROCESSES.. 524
B. W. Dickinson

THE BOUNDED p-POINT CLASSES IN ROBUST HYPOTHESIS TESTING AND FILTERING....................... 526
S. A. Kassam

OPTIMIZING THE MEASUREMENT PROCESSING ORDER FOR LINEAR REGRESSION WITH MULTIPLE INDEPENDENT VARIABLES... 535
T. G. Robertazzi and S. C. Schwartz

DETECTION AND ESTIMATION WITH FIXED LAG FOR ABRUPTLY CHANGING SYSTEMS........................ 545
V. J. Mathews and J. K. Tugnait

STATISTICAL EFFICIENCY OF THE SAMPLE AUTOCORRELATION FUNCTION IN ARMA PARAMETER ESTIMATION... 555
S. P. Bruzzone and M. Kaveh

III-D: COMPUTER ARCHITECTURE

Chairman: J. H. Patel
University of Illinois

REAL TIME SYSTEMS: AN APPROACH TO COMPUTER AIDED DESIGN OF HARDWARE AND SOFTWARE.................. 557
A. A. Ross, H. H. Loomis, Jr. and G. G. Pollock

A GRAPHICAL INTERCONNECTION LANGUAGE AND ITS APPLICATION TO CONCURRENT AND REAL-TIME PROGRAMMING..... 567
B. W. Weide, M. E. Brown, J. A. S. Alegria and G. Meyer

VARIETIES OF DECOUPLED ACCESS/EXECUTE COMPUTER ARCHITECTURES.................................. 577
J. E. Smith and T. J. Kaminski

EFFECTS OF MULTIACCESS MEMORY AND FAST MEMORY ON MULTIPROCESSING................................ 587
P. C.-C. Yeh

A VLSI-BUILDING BLOCK FOR THE PROCESSOR-TO-MEMORY COMMUNICATION IN LARGE-SCALE MULTIPROCESSOR SYSTEMS..... 589
B. Quatember

MEASURING PARALLELIZABILITY DEGREE OF PROGRAMS.......... 599
K. Culik and I. A. Natour

ON RUN-TIME PARALLELISM IN SEQUENTIAL PROGRAMS.......... 607
G. D. Lakhani

IV-A: VLSI

Chairman: D. J. Brown
University of Illinois

VLSI IMPLEMENTATIONS OF GEOMETRIC TASKS................ 615
B. Chazelle

SYSTOLIC MATRIX MULTIPLICATION ON A LINEAR ARRAY........ 625
I. V. Ramakrishnan, D. S. Fuzzell and A. Silberschatz

A NEW LAYOUT FOR CCC AND A NEW VLSI
STRUCTURE - CCC AND MESH.............................. 627
X. Jin

AN OPTIMAL ALGORITHM FOR OPTIMAL
ROUTING AROUND A RECTANGLE............................ 636
T. F. Gonzalez and S.-L. Lee

ANALYSIS OF HEURISTICS FOR PLA FOLDING.................. 646
S. S. Ravi and E. L. Lloyd

ON AN EFFICIENT IMPLEMENTATION OF A PLANARITY
TESTING ALGORITHM FOR A GRAPH WITH LOCAL CONSTRAINTS.... 656
K. Nakajima and M. Sun

MULTILECTIVE PLANAR CIRCUIT SIZE....................... 665
J. E. Savage

IV-B: CODING

Chairman: R. E. Blahut
IBM

HAMMING CODES, COMPUTER MEMORIES,
AND THE BIRTHDAY SURPRISE.............................. 672
R. M. F. Goodman and R. J. McEliece

BOUNDS TO THE RATE-DISTORTION FUNCTION
FOR REGENERATIVE COMPOSITE SOURCES..................... 680
M. J. Carter and D. L. Neuhoff

A DECODING SCHEME FOR THE TERNARY GOLAY CODE........... 682
V. Pless

AN OPTIMAL APPROACH FOR TIME-DOMAIN TREE
CODING OF STATIONARY GAUSSIAN SOURCES.................. 689
B. Mazor and W. A. Pearlman

CAPACITY AND CUTOFF RATE OF FREQUENCY-HOPPED
CHANNELS WITH SLOW RAYLEIGH FADING..................... 697
W. E. Stark

PARALLEL SPEECH ENCODING WITH MICROPROCESSORS.......... 704
S. Mohan, D. P. Agrawal and S. M. Komandur

IV-C: SIGNAL DETECTION

Chairman: G. L. Wise
University of Texas

MIN-MAX DETECTION OF A WEAK SIGNAL
IN STATIONARY MARKOV NOISE............................. 713
G. V. Moustakides and J. B. Thomas

M-STAGE NONPARAMETRIC RANK DETECTORS WHICH REDUCE THE AVERAGE SAMPLE SIZE AND RANKING TIME......... 723
S. Tantaratana

ON THE USE OF LIMITING IN CONJUNCTION WITH THE DETECTION OF SIGNALS IN PARTIALLY CONTAMINATED NOISE.... 733
D. R. Halverson

ON NARROWBAND IMPULSIVE NOISE............................ 739
K. S. Vastola

ASYMPTOTIC PERFORMANCE OF DETECTORS WITH NON-ZERO INPUT SNR.. 749
A. B. Martinez

ADAPTIVE DETECTION OF A KNOWN SIGNAL IN NON-GAUSSIAN NOISE...................................... 759
S. V. Czarnecki and J. B. Thomas

DETECTION OF SURE SIGNALS IN NOISE OF BOUNDED MAGNITUDE.. 769
Y. F. Huang

IV-D: OPTIMAL CONTROL AND LARGE SCALE SYSTEMS

Chairman: T. Basar
University of Illinois

PROBABILISTIC OBSERVATION STRATEGIES FOR SIMULTANEOUS DIFFERENTIAL GAMES WITH OBSERVATION COST... 779
Z. T. Zhu and R. W. Longman

ON THE LINEAR QUADRATIC CLOSED LOOP, NO MEMORY NASH GAME.. 788
G. P. Papavassilopoulos and G. J. Olsder

OPTIMAL DECENTRALIZED DISCRETE TIME SYSTEMS............ 791
K. Rahnamai and M. Sawan

OLIGARCHICAL CONTROL OF DISTRIBUTED PROCESSING SYSTEMS...................................... 793
M. Krieger and N. Santoro

ON SELECTION OF WEIGHTING MATRICES IN THE LOG PROBLEM... 803
R. E. Skelton and M. L. De Lorenzo

ACTUATOR PLACEMENT IN MODAL SYSTEMS USING FUEL OPTIMAL DEGREE OF CONTROLLABILITY CONCEPTS............. 813
R. A. Laskin, R. W. Longman and P. W. Likins

APPLICATION OF DISTURBANCE ACCOMMODATING CONTROL TO THE AUTOMATIC STEERING CONTROL OF A LARGE TANKER........ 823
R. E. Reid, B. C. Mears and A. K. Tugcu

V-A: SINGULAR PERTURBATION AND MODEL REDUCTION

Chairman: H. Khalil
Michigan State University

OBSERVABILITY AND IMPULSE OBSERVERS IN DESCRIPTOR-VARIABLE SYSTEMS........................ 825
D. Cobb

STRONG CONTROLLABILITY AND STRONG OBSERVABILITY OF GENERALIZED DYNAMICAL SYSTEMS...................... 834
G. Yan and T. J. Tarn

INFINITE-TIME REGULATORS FOR SINGULARLY PERTURBED DIFFERENCE EQUATIONS........................ 843
B. Litkouhi and H. K. Khalil

APPLICATIONS OF A STABILITY CRITERION FOR SINGULARLY PERTURBED SYSTEMS....................... 855
A. Saberi

BALANCED CONTROLLER REDUCTION.......................... 865
A. Yousuff and R. E. Skelton

PSEUDO-BALANCED REALIZATIONS FOR DETERMINISTIC AND STOCHASTIC DISCRETE-TIME SYSTEMS.................. 874
R. J. Vaccaro and B. W. Dickinson

AN ALGORITHM FOR SINGULAR VALUE DECOMPOSITION AND ITS APPLICATION TO LINEAR SYSTEM MODEL REDUCTION.. 884
M. S. Kamel and K. Singhal

DISCRETE-TIME SYSTEMS WITH SLOW AND FAST MODES......... 886
M. T. Tran and M. E. Sawan

V-B: COMBINATORIAL ALGORITHMS

Chairman: M. C. Loui
University of Illinois

ON THE OPTIMAL PLACEMENT OF RECOVERY POINTS............ 888
M.-J. Chung and M. S. Krishnamoorthy

USING DIVIDE AND CONQUER TO FIND FLOWS IN DIRECTED PLANAR NETWORKS IN $O(n^{3/2}\log n)$ TIME.................... 898
D. B. Johnson and S. M. Venkatesan

SOME ALGORITHMS FOR SCHEDULING INDEPENDENT VECTOR TASKS ON A MULTIPIPE VECTOR PROCESSOR.................. 906
S. K. Dhall and A. Dalili-Yazdi

MINIMAL SPANNING FORESTS IN BINARY TREES............... 914
F. Luccio and L. Pagli

ELABORATING ON THE BOYER-MOORE-GALIL STRING SEARCHING STRATEGIES............................ 924
A. Apostolico and R. Giancarlo

THE N-TH POWER OF A COMPANION MATRIX: FAST SOLUTIONS TO LINEAR RECURRENCES.................... 934
C. M. Fiduccia

HYBRID ALGORITHMS FOR NP-HARD PROBLEMS................. 941
D. K. Friesen and M. A. Langston

SEARCHING IN POLYNOMIAL AVERAGE TIME................... 943
P. Purdom and C. Brown

V-C: COMMUNICATION SYSTEMS

Chairman: J. G. Dunham
Washington University

MINIMAX ROBUST EQUALIZATION FOR RANDOM SIGNALS THROUGH UNCERTAIN CHANNELS...................... 945
G. Moustakides and S. A. Kassam

DESIGN OF SPREAD SPECTRUM SIGNAL SETS WITH SMALL MEAN-SQUARE CROSSCORRELATION................. 955
G. R. Cooper and E. H. Cooper

ON THE EVALUATION OF THE POWER SPECTRAL DENSITY OF TIME JITTER PRODUCED IN A LINE REGENERATOR.......... 957
S. Pupolin and C. Tomasi

ERROR-PROBABILITY BOUNDS FOR RANDOMLY DISPERSIVE OPTICAL CHANNELS........................... 965
J.-S. Cheng and D. L. Snyder

ON THE APPLICATION OF QUANTISED FEEDBACK AND PARTIAL RESPONSE TECHNIQUES TO HIGH BIT-RATE OPTICAL FIBRE COMMUNICATIONS.......................... 967
J. J. O'Reilly, M. Duarte and L. Blank

SIGNAL DESIGN FOR POWER EFFICIENT OPTICAL COMMUNICATION.. 975
D. L. Snyder and C. N. Georghiades

ON THE BEHAVIOR OF THE DATA QUEUE IN A VOICE/DATA INTEGRATED SYSTEM.......................... 977
D. D. W. Yao and A. Leon-Garcia

UPPER AND LOWER BOUNDS ON THE AVERAGE ERROR PROBABILITY FOR OPTICAL COMMUNICATION SYSTEMS.......... 985
J. R. F. da Rocha and J. J. O'Reilly

V-D: FAULT-TOLERANT SYSTEMS

Chairman: J. A. Abraham
University of Illinois

CHARACTERIZATION OF A HIGH AVAILABLE DUPLEX LARGE SCALE SYSTEM BY CELL PARTITION.................. 987
F. Lombardi and O. J. Davies

PERFORMANCE MEASURES FOR FAULT-TOLERANT MULTIPROCESSOR CONTROLLERS FOR CRITICAL PROCESSES....... 996
C. M. Krishna and K. G. Shin

A PROPOSAL FOR FAULT-TOLERANCE IN A MULTIPROCESSOR SYSTEM.................................... 998
P. Corsini, L. Lopriore and L. Strigini

REALIZING FAULT-TOLERANT BINARY TREES IN VLSI.......... 1008
P. J. Varman and D. S. Fussell

ON DIGRAPHS WITH MINIMUM DIAMETER AND MAXIMUM CONNECTIVITY.................................. 1018
S. M. Reddy, J. G. Kuhl, S. H. Hosseini and H. Lee

TRANSIENT ANALYSIS OF A PACKET SWITCHING NODE WITH A BUFFER ADMISSION SCHEME

G.B. AGNEW and J.W. MARK
Dept. of Electrical Engineering
University of Waterloo
Waterloo, Ontario, Canada N2L 3G1.

Traffic loads in packet switching networks are inherently bursty in nature, thus, steady state conditions are seldom approached. To observe the effects of sudden fluctuations in load, transient analysis must be performed. This will allow us to observe some of the time dependencies in the queueing network which cannot otherwise be obtained. Exact solutions for the transient behaviour of queues tends to be very cumbersome even for simple queueing models. Thus, we look at approximate methods for analysing non-equilibrium effects in queues which are more adaptable.

Diffusion approximations have been applied to certain queueing models in the past. Kobayashi [1],[2], developed a diffusion approximation for a queueing problem with both open and closed networks. The diffusion approximation is the solution to Kolmogorov's forward equation which defines a probability function $p(x_0,x;t)$. This function describes the probability of finding a queue of length x at time t given that it had an initial size of x_0 at $t=0$. In its most general form, Kolmogorov's forward equation can be written as

$$\frac{\partial p(x_o,x;t)}{\partial t} = \frac{1}{2}\frac{\partial^2}{\partial x^2}(\alpha(x,t)p(x_o,x;t) - \frac{\partial}{\partial x}(\beta(x,t)p(x_o,x;t))$$

where $\beta(x,t)$ and $\alpha(x,t)$ are the drift and variance terms respectively.

In our study, we consider a buffer admission scheme where traffic is divided into classes based on the radial distance to the message's destination. Blocking is imposed on the various classes if the total number of messages in the queue exceeds the prescribed buffer limit. This scheme is described in detail in our earlier studies [3],[4], and it is shown that this scheme will maintain throughput in the presence of heavy input overloads. We model the buffer admission scheme by considering a queue with a constant service rate μ and a state dependent arrival rate λ where

$$\lambda = \begin{matrix} \lambda_1 > \mu \quad \text{for } x < N \\ \lambda_2 < \mu \quad \text{for } x > N \end{matrix}$$

In our analysis, we have shown ,[5], that we can approximate a solution to the above model as a linear combination of the diffusion processes in each region (i.e. $x < N$ and $x > N$).

Results

Fig. 1 shows the results of the diffusion process at various instances in time. We note that the observed performance is what we intuitively expect for the model. To verify the accuracy of our approximation, numerous simulations were performed on the buffer admission model. These results were then compared to the diffusion approximation results. Fig. 2 shows a comparison of the results for $\lambda_1 = (1.75*\mu)$ and $\lambda_2 = (0.5*\mu)$ and a buffer boundary at $N = 20$. In the cases studied, a strong correlation in the results exists which supports the use of the approximation.

Thus we have demonstrated a technique based on a diffusion process which allows us to examine transient conditions in a queue with a buffer admission scheme. We have also developed a piecewise linear method for extending the results to higher class boundaries [5]. The results are significant in that we can predict the amount of overload traffic a queue can absorb before serious blocking occurs and, we can estimate the time required for the queue to establish a new equilibrium operating point in the presence of a continued overload.

References

1. Kobayashi, "Application of the diffusion approximation to queueing networks: Part I", Journ. of ACM, Vol. 21, No. 2, pp. 316-328, Apr. 1974.
2. Kobayashi, "Application of the diffusion approximation to queueing networks: Part II", Journ. of ACM, Vol. 21, No. 2, pp. 459-469, July 1974.
3. G.B. Agnew, J.W. Mark, 'On Distributed Network Routing and Buffer Management', Proceedings of NTC'80, Houston, Texas, pp. 59.3.1-5, Dec. 1980.
4. G.B. Agnew, J.W. Mark, 'Performance Modelling for Communication Networks with a Decision Node', Submitted to IEEE Trans. on Comm.
5. G.B. Agnew, 'Techniques for Network Buffer Management and Congestion Control', Ph.D. dissertation, Dept. of Electrical Engineering, University of Waterloo, May 1982.

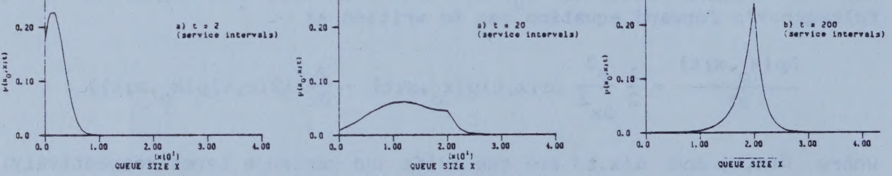


Figure 1 Diffusion Process for t = 2,20,200
λ_1=1.5µ, λ_2=0.5µ and N = 20

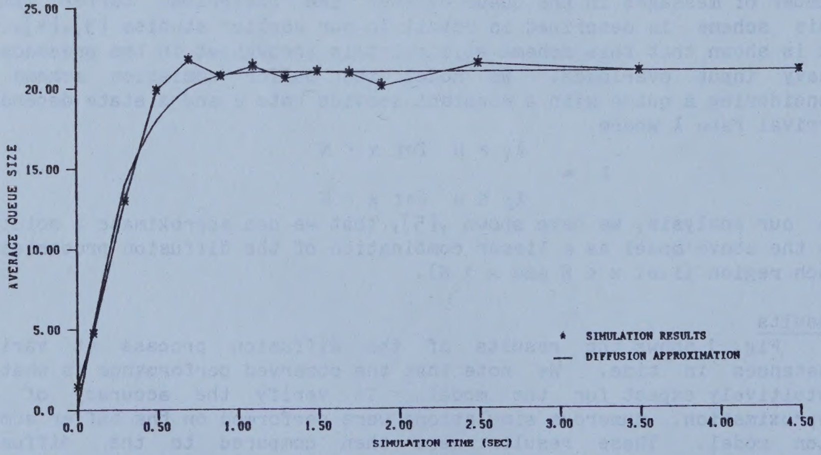


Figure 2 Comparison of Simulation Results and Diffusion Approximation Results for First Server (with Boundary)

Data Link Flow Control with Partial Observations

Aurel A. Lazar

Columbia University
Dept. of Electrical Engineering

Abstract

The modeling and data link flow control in computer communication networks is investigated. The receiving node is assumed to accept two classes of users. The first class models the packet flow from source to destination. These packets are subject to flow control. The second class models the interfering traffic due to the existing resource sharing with the traffic load resulting from other nodes in the network. The latter traffic, modeled as a continuous Poissonian flow of a fixed rate δ, is not directly observable. An optimal control strategy for the data link flow control protocol is found that maximizes the total average throughput under a bounded total average time delay criterion. This strategy uses partial observations: only the number of the first class of packets is available for controlling the data link. The resulting optimal control is shown to be a window flow control mechanism. The window size L is a function of the maximum tolerated time delay T, the input capacity c and the interfering packet flow δ.

I. Introduction

The problem of modeling and data link flow control in computer communication networks has been previously investigated [4], [6] (see also the references therein). The packet flow generated by the source is controlled using the information carried by acknowledgements. These acknowledgements are either piggy-backed by packets that are transmitted in the opposite direction or transmitted separately. In the following it is assumed that no time delay is incurred on the acknowledgement packets. Our methods leading to the optimal control strategy can easily be extended, however, to account for non-zero delays. As packets are forwarded they encounter other packets at the receiving node. These packets belong to other sources and their flows are modeled as an independent Poissonian stream with rate δ and having their own routing. The model above is therefore a mixed *BCMP* network [1].

In [6] (see also [7]) the interfering source-destination packet flow is modeled as a Poisson process and the probability of blocking is considered as a measure of congestion. By contrast, in this paper it is assumed that the interfering traffic flow due to the resource sharing with the other traffic is not directly observable. The aim is to control the flow of the first class of packets in the system in such a way that the total average throughput is maximized subject to an upper bound constraint on the total average time delay. In addition, the capacity of the input line is assumed to be c.

The research reported here was supported in part by the National Science Foundation under Grant # ECS-81-06476 and in part by the Office of Naval Research under Contract N00014-82-K-0508.

II. The Optimal Control

The queueing model for the data link flow control is depicted in Fig.1.

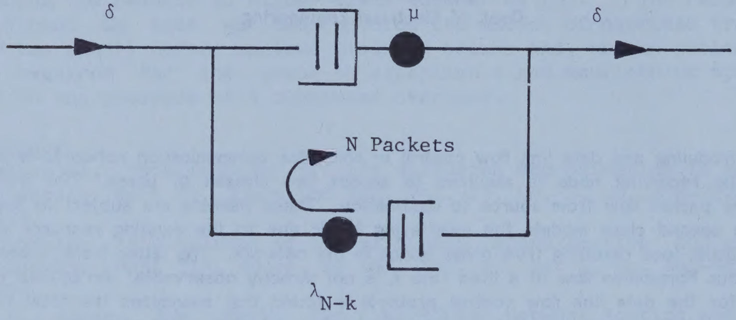


Fig. 1 A model for the data link flow control with partial observations.

Our main result is the following:

Theorem. *Given that* $T_{\max}^{(L-1)} < T \leq T_{\max}^{(L)}$, $2 \leq L \leq N$, *the end-to-end optimal flow control in equilibrium with a maximum of N controlled packets in the system is given by:*

$$\lambda_k = \begin{cases} c & N-L+2 \leq k \leq N \\ \lambda_{N-L+1} & k=N-L+1 \\ o & 1 \leq k \leq N-L \end{cases}$$

where

$$\lambda_{N-L+1} = \frac{T-\frac{1}{\mu-\delta}}{\frac{\rho+L}{\mu-\delta}-T} c\rho\left(\frac{c}{\mu-\delta}\right)^{1-L} + \frac{1}{\frac{\rho+L}{\mu-\delta}-T} \sum_{k=1}^{N} [(\mu-\delta)T-(\rho+k)]\left(\frac{c}{\mu-\delta}\right)^{k+1-L} .$$

Finally the total maximum average throughput amounts to:

$$F(T) = \mu(1-(1-\rho)\{1+\frac{T-\frac{1}{\mu-\delta}}{\frac{\rho+L}{\mu-\delta}-T}\rho$$

$$+\frac{1}{\rho+L-(\mu-\delta)T} \sum_{k=1}^{N} (L-k)\left(\frac{c}{\mu-\delta}\right)^{k}\}^{-1}) .$$

The interested reader can obtain a copy of the uncut version of the original manuscript by contacting the author.

ON A NEW LOCAL AREA NETWORK HYBRID ACCESSING SCHEME APPLYING SPLIT-CHANNEL ALLOCATION

BASILE DIMITRIADIS	NIKITAS ALEXANDRIDIS	GEORGE POLYZOS
Cleveland State Univ.	National Technical Univ.	National Technical Univ.
Cleveland, Ohio	Athens, Greece	Athens, Greece

ABSTRACT

The problem of a local area computer network hybrid accessing scheme is examined, where two groups of users are competing. Each group presents different traffic characteristics (the first, steady heavy traffic and the second, light bursty traffic). Two different accessing schemes were used simultaneously (TDMA and CSMA for each group respectively), utilizing independent subchannels by an FDM partitioning of the total bandwidth. An operating cost optimality criterion was used for the bandwidth allocation and the system performance was evaluated for various user population scenarios. An implementation concept of the proposed scheme is finally outlined.

1. INTRODUCTION AND PROBLEM STATEMENT

A major problem to consider in the design of large computer communications networks serving subscribers of different traffic characteristics, is to wisely allocate the network resources so that a performance criterion is optimized.

It is known in this context ([1],[2]), that for users of widely different behavior, a single network operating mode cannot be optimally applied, and it has been found ([3], [4]) that random access modes are well suited to bursty light-average traffic users, while fixed-assignment access modes serve better uniform heavy-traffic users.

It is therefore expected that a hybrid network accessing scheme, accommodating both the above modes with properly optimized resource-sharing parameters, will perform and serve in a favorable way with different user-population composition scenarios.

For this purpose, one must define the environment of the network and the performance criteria under consideration.

A Local Area Network architecture is considered with a high-bandwidth common data highway, where each user is attached through a common standard interface, controlled by a microprocessor.

Assuming constant total cable capacity (bandwidth W), we apply an FDM-type bandwidth partitioning into two subchannels W_1 and W_2. Each one serves independently a TDMA-type protocol (W_1) and a CSMA-type protocol (W_2), for the heavy and steady as well as for the light and bursty traffic users respectively. Another subchannel, W_c, is also needed for control purposes proved subsequently to be of a much smaller capacity than the main subchannels. Obviously, $W_1 + W_2 + W_c = W$. An optimal allocation of W_1 and W_2 is sought., when user population data and traffic offered are given.

It is explicitly assumed that the statistics of data-packet generation and the packet characteristics are known for each user and that actually only two categories of similar users are initially admitted, one with high steady traffic and one for low bursty traffic.

Poisson statistics are assumed for the packet generation with rates of λ_i packets/user sec (i=1,2), and packet lengths b_i (bits) depending upon the category.

It is a common practice in the analysis of network accessing schemes (e.g. [5], [7]) to express the performance and load characteristics as normalized magnitudes over the packet length (or duration). However, since the scheme examined here exhibits variable subchannel capacities (in bits/sec) and different packet sizes (bits), such a uniform normalization is impossible and performance measures such as average throughput or average time delay per packet served are either ill-defined or difficult to use.

A more universal performance measure must be used and system optimization conditions must be examined in order to accommodate variable user characteristics, as the proposed scheme dictates.

Also, for a fast and stable adaptation to variable conditions, proper control algorithms have to be investigated.

In order to achieve computational efficiency and speed, a central controller monitoring the overall system operation and traffic statistics and realizing the optimization and control algorithms must be implemented. Communication of the optimal control parameters to the participating users is performed through the control subchannel W_c. A distributed realization of these tasks can also be examined.

The problem, therefore, that is handled in this paper can be summarized as follows:

> Given a total channel bandwidth W, serving two categories of users as indicated, optimally partition it into two completely independent subchannels W_1 and W_2, one serving a TDMA protocol (W_1) and the other a CSMA-CD protocol (W_2), so that given the total number of users in each category m_1 and m_2, a performance criterion will be optimized. The small control subchannel, W_c, is not included in W and is considered separately for simplicity.

Such a performance criterion has been selected to be the overall cost related to the network traffic (per user-category), the user priority policy, the bandwidth allocations per user-category, and the delay per bit of traffic offered. In the next paragraph, the cost function is described and a minimization procedure is developed.

Numerically obtained simulation results have shown that the cost-function chosen is convex over the bandwidth space (W_1, W_2) and that a minimization is possible under a very broad set of assumptions. Figure 1 summarizes the proposed hybrid accessing scheme.

2. THE HYBRID ACCESSING SCHEME

As indicated in paragraph 1, a TDMA protocol has been used for channel W_1 serving m_1 users each offering a λ_1 Poisson rate of packets - each one being b_1 bits long. We assume that each user has an infinite buffer where he stores the packets ready for servicing by the network.

It is well known [1] that the average delay per packet under the above scheme is

$$d_1(W_1) \triangleq d_1 = \frac{\rho^2/\lambda_1}{2(1-\rho)} + \frac{m_1 b_1}{2W_1} + \frac{b_1}{W_1} \quad (\text{sec}) \tag{1}$$

where $\rho = m_1 \dfrac{\lambda_1 b_1}{W_1}$

Similarly, a CSMA-protocol has been used for subchannel W_2, serving m_2 users each offering a λ_2 Poisson rate of packets, b_2 (bits) long. The average delay per packet is then [1]:

$$d_2(W_2) \triangleq d_2 = \frac{b_2}{W_2} \left\{ \left(\frac{G}{S} - 1\right)(1+\alpha+k) + 1 + \alpha \right\} \tag{2}$$

where

$$G = \lambda_2 m_2, \quad \alpha = \frac{\tau}{T} = \frac{\tau W_2}{b_2}, \quad k = \frac{\overline{x}\, W_2}{b_2}, \quad S_2 = \frac{Ge^{-\alpha G}}{G(1+2\alpha) + e^{-\alpha G}}$$

and τ the maximum signal propagation time through the medium (cable), while $\overline{x}$ is the average packet retransmission time after collision, depending upon the underlying statics assumed.

The optimization criterion chosen in this work is as mentioned before, the overall operating cost of the network given by the following equation:

$$C(W_1,W_2) = c_1 m_1 b_1 d_1 (W_1) \lambda_1 + c_2 m_2 b_2 d(W_2) \lambda_2 \tag{3}$$

where c_1 and c_2 is the unit cost for each subchannel (in 1/bit.normalized) and d_1, d_2 taken from equations (1) and (2), while m_1, m_2, b_1, b_2 are already defined above.

Now expressing $W_2 = W-W_1$ and differentiating (3) with respect to W_1, one can obtain conditions of minimization of $C(W_1,W_2)$ with respect to W_1. One can show that $C(W_1,W-W_1)$ is a convex function and that $\frac{\partial C}{\partial W_1} = 0$ always gives a minimum, provided that $m_1 m_2 \neq 0$ (trivial) ([8] and [9]). Figure 2 presents $C(W_1,W_2)$ as obtained by computer simulation.

The analytical solution of $\frac{\partial C}{\partial W_1} = 0$ is extremely difficult so numerical methods have been applied, resulting in the determination of the optimal value of W_1, let's call it W_1^*. Then $W_2^* = W - W_1^*$ is the respective optimal value of the CSMA subchannel bandwidth. By determining the optimal allocation of total channel bandwidth, given the user population (per category) m_1, m_2 and the underlying traffic statistics λ_i, b_i the problem is considered solved. The central control facility of the network undertakes the solution of the optimization problem for every given user scenario (including m, b, λ and c's).

Since, however, the TDMA protocol is a synchronous accessing scheme and very strict timing constraints have to be imposed, (the CSMA subchannel is relatively insensitive to timing deviations), a universal timing facility is required with frequent user clocks synchronizations. The sync information, together with the optimal W_1^* and W_2^* values are communicated to the participating users via the W_c subchannel.

3. SOME NUMERICAL RESULTS

In the research conducted, numerical results have been obtained for scenarios including both the bursty and steady traffic user populations, competing for access of the common medium (cable).

Different population ratios have been examined and different user statistics have been considered.

It has been found that the optimality criterion chosen (overall cost), is rather insensitive to a large variation of W_1/W ratio around its minimum point (Figure 2). However, it has also been demonstrated that under widely varying m_1/m_2 ratios (user populations ratios), the proposed scheme performs better with respect to both the average cost and the average delay, compared to the cases that pure CSMA or TDMA techniques were used to serve the same traffic loads under the same conditions. Figures 3 and 4 indicate this.

It has also been demonstrated that as the overall channel capacity is increasing, the hybrid accessing scheme gives still less total cost and average delay compared to CSMA or TDMA alone, for various bursty to steady traffic ratios (Figures 5 and 6).

These results indicate that the proposed scheme is a well-behaving network accessing scheme, able to accommodate a widely varying set of users exhibiting large differences in their traffic statistics. It can also be shown that the adaptation characteristics of the scheme to sudden changes of traffic state (number of users, or packet loads, etc.) are very good and that the system promptly reallocates its resources to optimize the cost under the new circumstances.

4. PROPOSED SCHEME REALIZATION

In Figure 7, a model network topology is presented, including the central controller and the participating user interfaces (Figure 8). Each interface includes a physical link connection including both the cable tap and the FDM stage (channel partitioning hardware together with the T_x/R_x stages for each operating mode), a physical user interface connection and a microcomputer implemented control module.

The scope of this module is to decode the central controller's information about the W_1^* and W_2^*, to adjust accordingly the FDM hardware, and to channelize the user offered traffic to the appropriate assigned subchannel. It is therefore indicated here that the same interface can be used for both the TDMA and the CSMA type users, according to the characteristics that each one exhibits and the minimization of cost function that the central controller implements.

An efficient network adaptation is then achieved leading to a very powerful and versatile hybrid accessing scheme. It is obvious that with the expense of some hardware redundancy, the optimization-algorithm implementation can be distributed and delegated to the peripheral smart user interfaces, achieving thus a completely decentralized protocol. Such work is currently under study and it will be reported on in the future.

5. CONCLUSIONS

An efficient hybrid accessing scheme has been proposed, able to accommodate bursty and steady-traffic users simultaneously, by dynamically allocating the common channel bandwidth into two subchannels, each serving a TDMA and a CSMA protocols independently, and through them the proper user category respectively.

Analytical results have been indicated, supporting the scheme's optimality characteristics.

An implementation topology has been proposed realizing the scheme presented here, using a central controller and uniform standardized user interfaces, both for the bursty and the steady traffic users.

From the analysis, it has been found that the optimality criterion used (average cost per bit of offered traffic), is not very sensitive to the selection of the optimal bandwidth partitioning point, especially in the CSMA area. This indicates that under the assumed circumstances, when the available channel bandwidth is very wide, even the non-bursty traffic users can be well accommodated by a random access protocol up to a saturation capacity.

The inherent flexibility, however, of the proposed realization allows the evaluation of the optimal bandwidth allocation policy and guarantees that the assumed performance metric will be minimized. A centralized implementation of the control scheme has been proposed with the purpose of evaluation of each user's statistics and the realization of the centrally obtained optimal control parameters, delegated to the "smart" user-network interfaces. This indicates a hierarchical control system, able to accommodate a distributed architecture. Research is currently underway in this direction.

BIBLIOGRAPHY

[1] F. Tobagi and L. Kleinrock, "Packet Switching in Radio Channels: Part III - Polling and (Dynamic) Split-Channel Reservation Multiple Access," IEEE Trans. on Communications, Vol. COM-24, pp. 832-844. August, 1976.

[2] J. Wieselthier and A. Ephremides, "A New Class of Protocols for Multiple Access in Satellite Networks," IEEE Trans. of Automatic Control, Vol. AC-25, pp. 865-879, October, 1980.

[3] N. Abramson, "The Throughput of Packet Broadcasting Channels," IEEE Trans. on Communications, Vol. COM-25, pp. 148-157, January, 1977.

[4] S. Lam, "Delay Analysis of a TDMA Channel," IEEE Trans. on Communications, Vol. COM-25, pp. 1484-1494, December, 1977.

[5] L. Kleinrock, "Queing Systems, Vols. I, II," John Wiley, New York, 1975, 1976.

[6] M. Ferguson, "A Bound and Approximation of Delay Distribution for Fixed-Length Packets in an Unslotted ALOHA Channel and a Comparison with TDM," IEEE Trans. on Communications, Vol. COM-25, pp. 136-139, January, 1977.

[7] G. Polyzos, "On a New Multiple Channel Access Scheme for Local Area Networks," Diploma of Electrical Engineering Thesis, National Technical University of Athens, Greece, June, 1982.

[8] B. Dimitriadis, N. Alexandridis and G. Polyzos, "Bandwidth Assignment in Local Data Communication Networks as a Function of Traffic Characteristics," to be submitted to IEEE Trans. on Communications.

[9] A. Ephremides and O. Mowafi, "Analysis of a Hybrid Access Scheme for Buffered Users-Probabilistic Time Division," IEEE Trans. on Software Engineering, Vol. SE-8, No. 1, pp. 52-61, January, 1982.

HYBRID ACCESSING SCHEME OVERVIEW

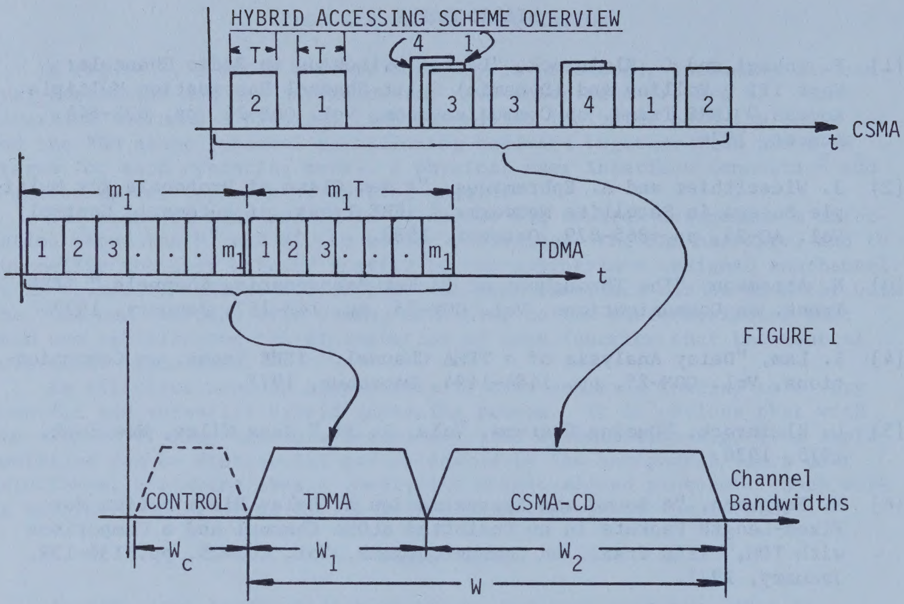

T
T
4
1
2
1
3
3
4
1
2
t
CSMA
m_1T
m_1T
1
2
3
.
.
.
m_1
1
2
3
.
.
.
m_1
TDMA
t
CONTROL
TDMA
CSMA-CD
Channel Bandwidths
W_c
W_1
W_2
W

FIGURE 1

COST FUNCTION VS. W_1/W RATIO

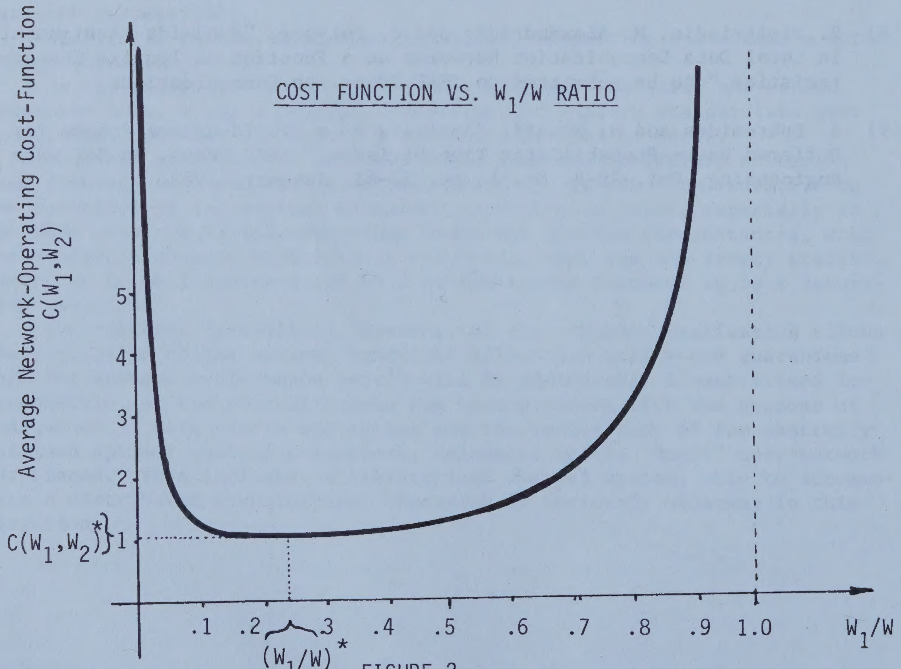


FIGURE 2

OPERATING COST VS. BURSTY USERS NUMBER
FOR CSMA, TDMA, AND HYBRID SCHEMES (m_1 = 10)

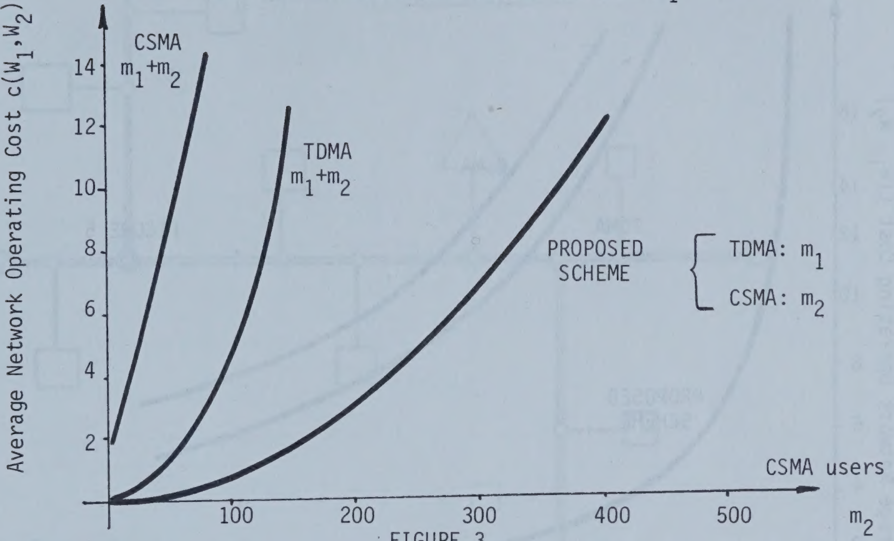


FIGURE 3

AVERAGE DELAY PER BIT VS. BURSTY USERS NUMBER
FOR CSMA,TDMA, AND HYBRID SCHEMES (m_1 = 10)

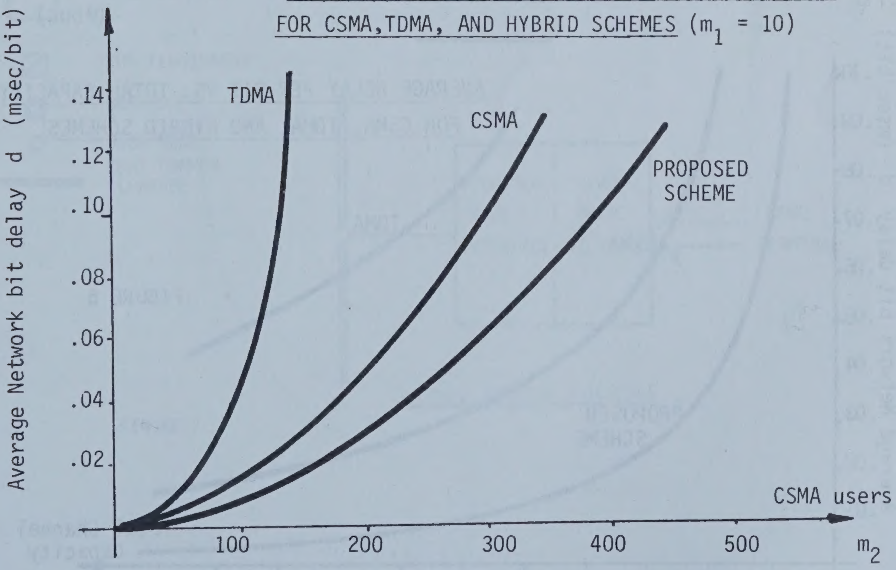


FIGURE 4

OPERATING COST VS. TOTAL CHANNEL CAPACITY
FOR CSMA, TDMA, AND HYBRID SCHEMES

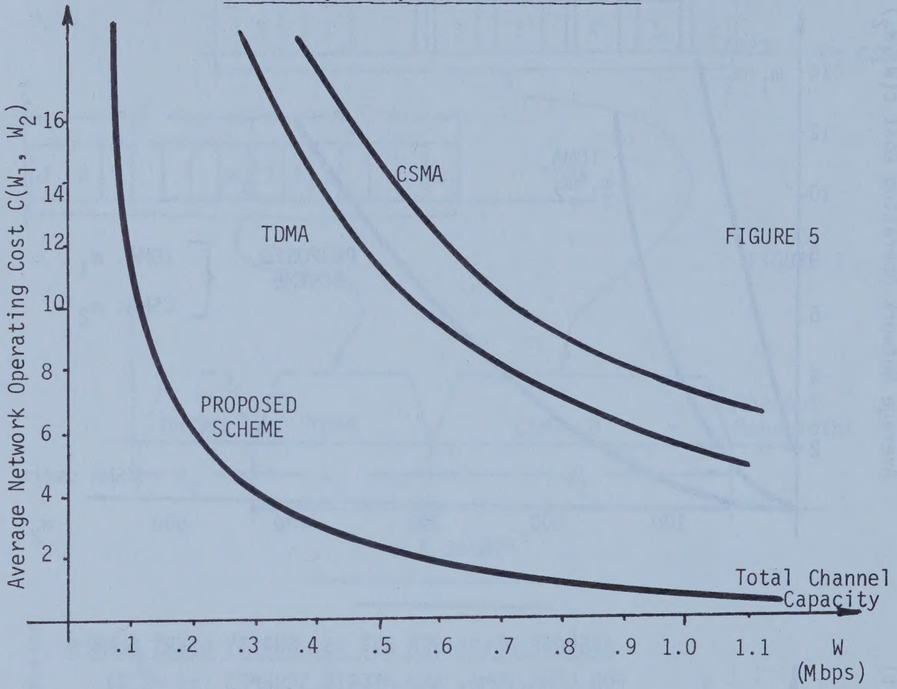


FIGURE 5

AVERAGE DELAY PER BIT VS. TOTAL CAPACITY
FOR CSMA, TDMA, AND HYBRID SCHEMES

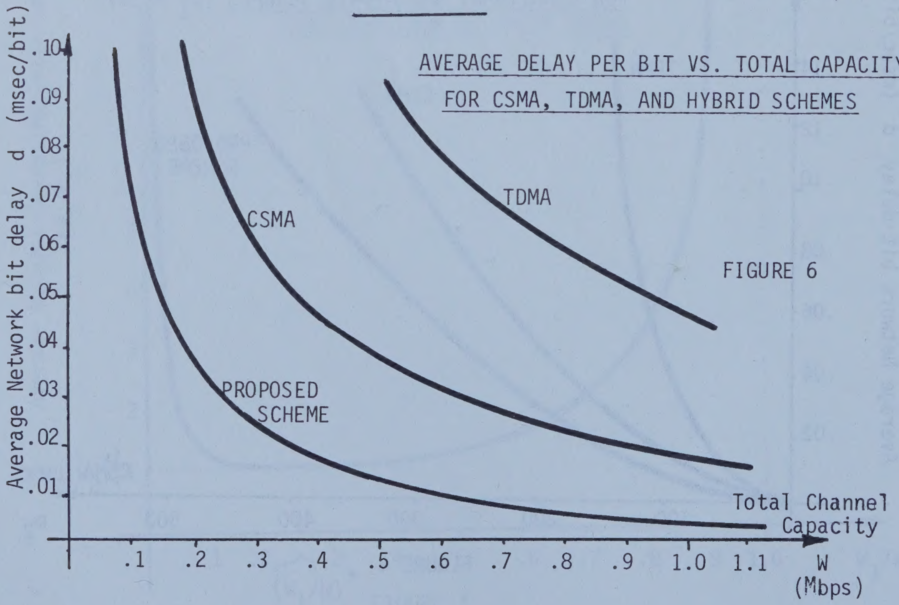


FIGURE 6

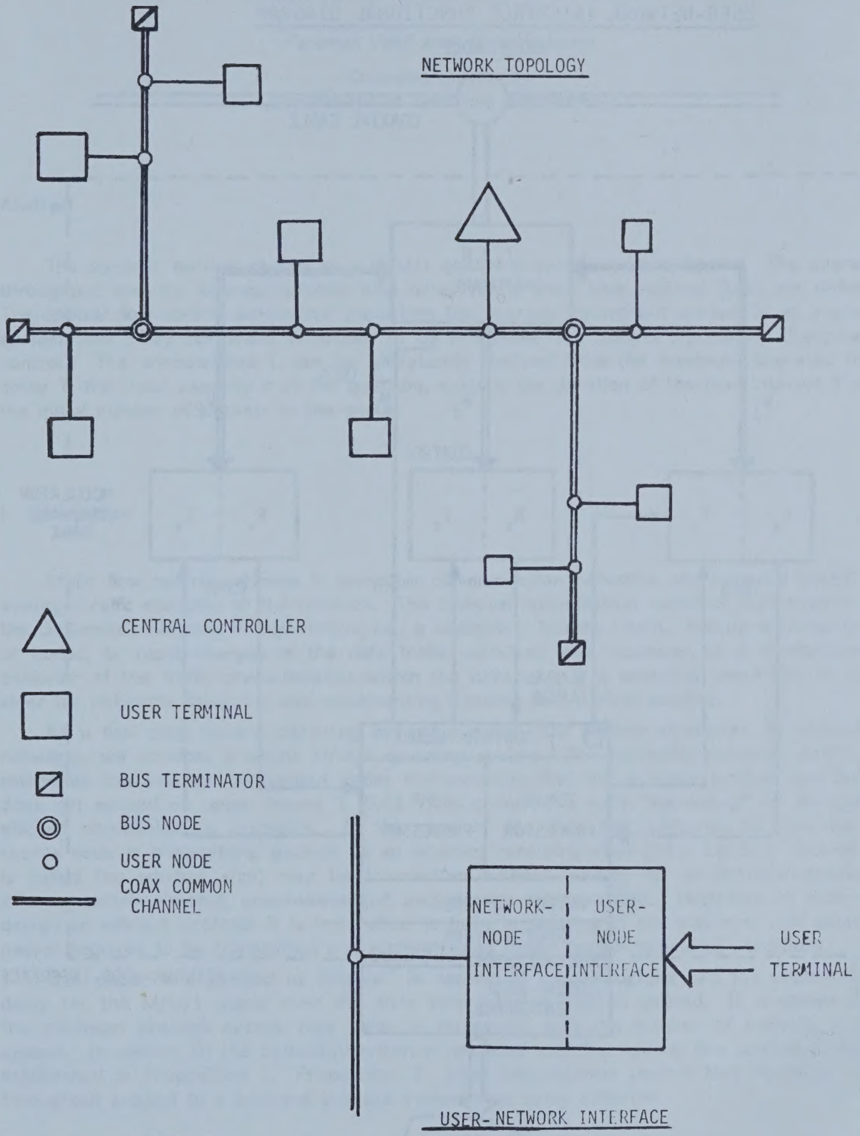

NETWORK TOPOLOGY
CENTRAL CONTROLLER
USER TERMINAL
BUS TERMINATOR
BUS NODE
USER NODE
COAX COMMON CHANNEL
NETWORK-NODE INTERFACE
USER-NODE INTERFACE
USER TERMINAL
USER-NETWORK INTERFACE

FIGURE 7

USER-NETWORK INTERFACE FUNCTIONAL DIAGRAM

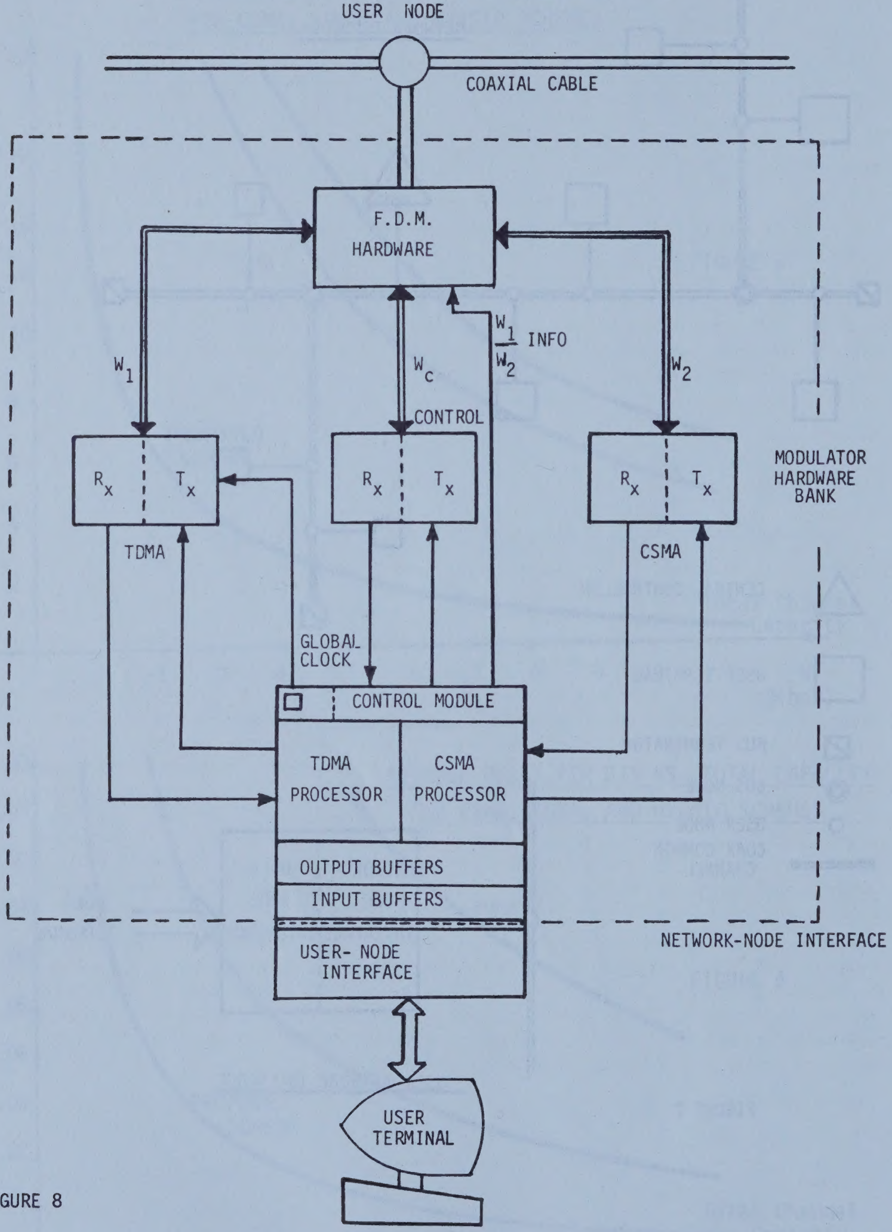

USER NODE
COAXIAL CABLE
F.D.M.
HARDWARE
W_1
W_c
$\frac{W_1}{W_2}$ INFO
W_2
CONTROL
R_x
T_x
R_x
T_x
R_x
T_x
MODULATOR
HARDWARE
BANK
TDMA
CSMA
GLOBAL
CLOCK
CONTROL MODULE
TDMA
PROCESSOR
CSMA
PROCESSOR
OUTPUT BUFFERS
INPUT BUFFERS
USER- NODE
INTERFACE
NETWORK-NODE INTERFACE
USER
TERMINAL

FIGURE 8

Dynamic Optimal Control of a M/M/1 Queue

Faramak Vakil and Aurel A. Lazar
Columbia University
Department of Electrical Engineering

Abstract

The dynamic optimal control of a M/M/1 queueing system is investigated. The average throughput and the average system time delay for a finite time interval [0,S] are defined. The optimal flow control policy that maximizes the average throughput subject to an average system time delay constraint is shown to be a window flow control mechanism (bang-bang control). The window size L can be analytically derived from the maximum tolerated time delay T, the input capacity c of the queueing system, the duration of the time interval S and the initial number of packets in the queue.

I. Introduction

Static flow control schemes in computer communication networks are designed based on average traffic statistics of the network. The classical mathematical model of such systems is the Jacksonian network in equilibrium, *i.e.*, a stationary Markov chain. Failure of some links or nodes, or rapid changes in the data traffic workload lead, however, to a nonstationary behavior of the traffic characteristics within the network. It is essential, therefore, to consider for networks designing and implementing dynamic flow control policies.

As a first step toward obtaining optimal dynamic flow control strategies for computer networks, we consider a simple M/M/1 queueing system. The optimality criterion employed maximizes the average throughput under the condition that the average system time delay does not exceed an upper bound T [5,6]. This criterion is very "appealing" for designing efficient communication protocols. In the case of the data link protocol, for example, a source node is transmitting packets to an adjacent receiving node [11]. Up to L packets (L is called the window size) may be transmitted without waiting for an acknowledgement. After a certain timeout, unacknowledged packets are retransmitted. Therefore in order to design an efficient protocol it is imperative to have knowledge of the maximum time delay a packet requires to be transmitted and acknowledged under normal operating conditions.

This paper is organized as follows. In section II the throughput and the system time delay for the M/M/1 queue over the finite time interval [0,S] is defined. It is shown that the maximum average system time delay is increasing with the number of packets in the system. In section III the optimality criterion is stated and the optimal flow control problem established in Proposition 1. Proposition 2 gives the optimum control that maximizes the throughput subject to a bounded average system time delay criterion.

II. The Throughput and System Time Delay of a M/M/1 Queue

Let us consider the queueing system depicted in Fig.1 . The upper queue with an exponential server μ has to be controlled over the finite time interval [0,S] corresponding to a suitable optimality criterion. The feedback queue has an unknown exponential server $\lambda(Q_t)$, where Q_t is the number of packets in the queue at time t, $0 \leq t \leq S$. Since there are a maximum number of N packets in the system, the upper queue can be seen as a M/M/1 queueing system with a finite buffer size N. Without any loss of generality, therefore, the feedback queue models the input stream to a M/M/1 system with finite buffer size. The above setting represents a simple model for the data link flow control between two adjacent nodes in a computer network where each packet is acknowledged upon its arrival [11]. The upper queue models the receiving node and the lower queue the source node. The time delay incurred on the acknowledgements is assumed to be negligible. For the modeling purpose it is also assumed that the source node has an infinitely large pool of packets.

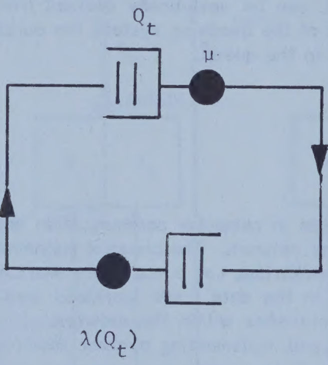


Fig.1 A model for the data link flow control

Let $Q=(Q_t,F_t)$, $0 \leq t \leq S$, be a Markov process with state space $\{0,1,\ldots,N\}$ and infinitesimal generators $\lambda=(\lambda(Q_t),F_t)$, $0 \leq t \leq S$, and μ, defined on the complete probability space (Ω,F,P). $Q=(Q_t,F_t)$, $0 \leq t \leq S$, is the stochastic process modeling the upper queue. Thus:

$$Q_t = Q_0 + A_t - D_t \quad , \tag{1}$$

where A_t and D_t are the total number of arrivals and departures over the time interval [0,t], respectively. Since:

$$D_S - \int_0^S \mu_u 1(Q_u > 0)du \quad ,$$

is a martingale [2], the average number of departures in [0,S] is given by:

$$ED_S = E\int_0^S \mu_u 1(Q_u > 0)du \quad .$$

With $\mu_u=\mu$, $u\in[0,S]$, the average throughput, *i.e.*, the average number of packets leaving the upper queue in the time interval [0,S] is given by:

$$E\gamma_S=\frac{1}{S}ED_S=\frac{\mu}{S}E\int_0^S 1(Q_u>0)du \quad ,$$

or

$$E\gamma_S=\frac{\mu}{S}\int_0^S E\,1(Q_u>0)du=\frac{\mu}{S}\int_0^S(1-p_0(u))du \quad ,$$

where $p_0(u)$ is the probability that the upper queue is empty at time u, $0\le u\le S$. Thus the average throughput is

$$E\gamma_S=\mu-\frac{\mu}{S}\int_0^S p_0(u)du \quad . \tag{2}$$

In what follows the notion of system time delay will be introduced. The average system time delay reflects the expected amount of time a packet spends in the system. The definition of the system time delay given below is motivated by the definition of the average time delay for a M/M/1 queueing system in equilibrium [5]. Let σ_S denote the total time delay of the packets arriving during the time interval [0,S].

Definition 1. *The system time delay* τ_S *of the packets arriving in the time interval* [0,S] *is given by :*

$$\tau_S=\frac{\sigma_S}{EA_S} \quad .$$

From the above definition one easily finds that the average system time delay amounts to:

$$E\tau_S=\frac{E\sigma_S}{EA_S} \quad .$$

The average total time delay for arrivals during the interval [0,S] can be written as [3]:

$$E\sigma_S=E\int_0^S Q_u du+\frac{1}{2\mu}E(Q_S^2+Q_S)-\frac{1}{2\mu}E(Q_0^2+Q_0) \,.$$

The second term on the right hand side of the above equality is the average total excess time delay of the packets that are in the system at time t=S . The third term is the average total excess time delay of the packets which were in the system at time $t=0$. The above expression can also be rewritten as:

$$E\sigma_S=\sum_{k=1}^{N}k\int_0^S p_k(u)du+\frac{1}{2\mu}\sum_{k=1}^{N}(k^2+k)(p_k(S)-p_k(0)) \quad ,$$

where $p_k(u)$ is the probability that the upper queueing system has k packets at time

u, $0 \leq u \leq S$. From (1) we have:

$$EA_S = EQ_S - EQ_0 + ED_S \ ,$$

and therefore

$$EA_S = \sum_{k=1}^{N} k(p_k(S)-p_k(0)) + \mu \sum_{k=1}^{N} \int_0^S p_k(u)du \ .$$

Hence the average system time delay is given by:

$$E\tau_S = \{ \sum_{k=1}^{N} k\int_0^S p_k(u)du + \frac{1}{2\mu} \sum_{k=1}^{N} (k^2+k)(p_k(S)-p_k(0)) \} \{ \mu \sum_{k=1}^{N} \int_0^S p_k(u)du + \sum_{k=1}^{N} k(p_k(S)-p_k(0)) \}^{-1} \quad (3)$$

Definition 2. $\lambda = (\lambda(Q_t), F_t)$, $0 \leq t \leq S$, *will hereafter denote the control.*

We now formally introduce the class of admissible controls.

Definition 3. The class of controls $\lambda = (\lambda(Q_t), F_t)$, $0 \leq t \leq S$, *such that* $0 \leq \lambda(Q_t) \leq c$, *for all* t, $0 \leq t \leq S$, *where* c, $c \in R_+$, *is a constant is called admissible.*

In order to simplify the notation let us set $\lambda(Q_t = k) = \lambda_k(t)$. The (N+1)x1 probability vector $P(t) = [p_k(t)]$ satisfies the Chapman - Kolomogorov equation:

$$\dot{P}(t) = A(\lambda)P(t) \ ,$$

where $A(\lambda) = [a_{ij}]$ is a (N+1)x(N+1) matrix with elements:

$$\begin{array}{llllll} a_{00} = \lambda_0(t) \ , & & a_{i,i+1} = \mu \ , & 0 \leq i < N \ , \\ a_{ii} = -\mu - \lambda_i(t) \ , & 1 \leq i < N \ , & a_{i,i-1} = \lambda_{i-1}(t) \ , & 1 \leq i \leq N \ , \\ a_{NN} = -\mu \ , & & a_{ij} = 0 \ , & i-j \geq 2 \ . \end{array}$$

Therefore, from (3) the average system time delay can be rewritten as:

$$E\tau_S = \frac{\int_0^S ZP(u)du}{\int_0^S YP(u)du} \ , \qquad (4)$$

where $Y = [y_k]$ and $Z = [z_k]$ are 1x(N+1) row matrices with elements given by:

$$y_k = \begin{cases} \lambda_k(t) & 0 \leq k \leq N-1 \\ 0 & k = N \end{cases}$$

and

$$z_k = \begin{cases} (k+1)\dfrac{\lambda_k(t)}{\mu} & 0 \le k \le N-1 \\ 0 & k=N \end{cases},$$

respectively. Thus, the system time delay amounts to:

$$E\tau_S = \frac{\frac{1}{\mu}\sum_{k=0}^{N-1}(k+1)\int_0^S \lambda_k(u)p_k(u)du}{\sum_{k=0}^{N-1}\int_0^S \lambda_k(u)p_k(u)du}, \tag{5}$$

or finally,

$$E\tau_S = \frac{1}{\mu} + \frac{\sum_{k=0}^{N-1} k\int_0^S \lambda_k(u)p_k(u)du}{\mu\sum_{k=0}^{N-1}\int_0^S \lambda_k(u)p_k(u)du}. \tag{6}$$

The first term in (6) is the average service time of a packet and the second term represents the average system time delay in the queue.

Remark: It can easily be seen that when S approaches infinity and the queueing system in Fig. 1 is in equilibrium the throughput and the average system time delay are given by:

$$\lim_{S\to\infty} E\gamma_S = \mu(1-p_0),$$

and

$$\lim_{S\to\infty} E\tau_S = \frac{\sum_{k=1}^{N} kp_k}{\mu\sum_{k=1}^{N} p_k}.$$

The above values are exactly the throughput and the average time delay for a M/M/1 queueing system in equilibrium [5].

With L packets present in the system, the L'th packet has to wait for the service completion of L-1 packets. Since the average service time for each packet is $\frac{1}{\mu}$, the maximum waiting time for the L'th packet will be $\frac{L}{\mu}$. This is proven in the following:

Lemma 1. *Let* $T_{\max}^{(L)}$ *denote the maximum average system time delay achieved with L packets in the system. Then*

$$T_{\max}^{(L)} \le \frac{L}{\mu}.$$

Proof: The maximum average system time delay with L packets in the system is achieved when $\lambda_k(t)=c$ for all k, $0 \le k < L$ and t, $0 \le t \le S$. Therefore from (5) we have :

$$T_{\max}^{(L)}=\frac{\frac{c}{\mu}\sum_{k=0}^{L-1}(k+1)\int_0^S p_k(u)du}{c\sum_{k=0}^{L-1}\int_0^S p_k(u)du}\leq\frac{L}{\mu} \quad ,$$

or

$$\sum_{k=1}^{L-1}(k+1-L)\int_0^S p_k(u)du\leq 0 \quad .$$

The latter inequality is always satisfied since $k\leq L-1$.

Remark: Given that $\lambda_k(t)=c$, for all k, $0\leq k<L$ and t, $0\leq t\leq S$, explicit expressions for the $p_k(t)$'s are available [9,10]. Therefore, the exact value of $T_{\max}^{(L)}$ can be easily calculated.

Lemma 2. *The maximum system time delay in the* M/M/1 *queueing system is strictly increasing with the number of packets L, i.e.,*

$$T_{\max}^{(L)}<T_{\max}^{(L+1)} \quad ,$$

for all L, L<N.

III. Dynamic Optimal Control of the M/M/1 Queue

Let us consider again the queueing model for the data link flow control given in Fig.1. The optimality criterion employed for controlling the upper queue maximizes the throughput subject to an average system time delay constraint.

Definition 4. The control $\lambda=(\lambda(Q_t),F_t)$, $0\leq t\leq S$, *is said to be optimum over the class of admissible controls for a given T,* $T\in R_+$, *if the maximum*

$$\max_{E\tau_S\leq T} E\gamma_S$$

is achieved.

Let Λ be the N-dimensional manifold defined by the set of all $(\lambda_k(t))$, $0\leq t\leq S$, $0\leq k<N$, such that the following constraints are satisfied:

$$0\leq\lambda_k(t)\leq c \quad ,$$

for all t, $0\leq t\leq S$, and

$$\int_0^S G'P(u)du\leq 0 \quad ,$$

where $G=[g_k]$ is a (N+1)x1 vector with elements

$$g_k = \begin{cases} \lambda_k(t)(\frac{k+1}{\mu} - T) & 0 \le k \le N \\ 0 & k=N \end{cases} .$$

From (2) it is obvious that in order to maximize the throughput it is necessary and sufficient to minimize the expression

$$\int_0^S p_0(u)du \quad .$$

The average system time delay constraint on the other hand is equivalent to:

$$\int_0^S G'P(u)du \le 0 \quad .$$

Therefore, we have proved the following:

Proposition 1. *$\lambda=(\lambda(Q_t),F_t), 0 \le t \le S$, is an optimum control if it achieves the minimum of*

$$\int_0^S BP(u)du \quad ,$$

subject to:

$$\dot{P}(t) = A(\lambda)P(t) \quad , \quad P(0)=P_0 \; ,$$

where $\lambda \in \Lambda$ and B = [1, 0, ... , 0] *is a* 1x(N+1) *row matrix.*

We can now give the following:

Proposition 2. *Given that $T_{\max}^{(L-1)} < T \le T_{\max}^{(L)}$ the optimum control $\lambda=(\lambda(Q_t),F_t)$, $0 \le t \le S$, is given by:*

$$\lambda(Q_t) = \begin{cases} c & 0 \le Q_t \le L-2 \\ \lambda_{L-1} & Q_t = L-1 \\ 0 & L \le Q_t < N \end{cases} .$$

Proof: Let us recall Pontryagin's necessary conditions for the existence of an optimum control policy [4, 5, 8] (see also [1]).

The Hamiltonian for the above system is given by:

$$H(P(t),\lambda(t),\alpha(t)) = BP(t) + \alpha'(t)A(\lambda)P(t) \quad .$$

The optimal control satisfies the state and costate equations:

$$\dot{P}(t) = \frac{\partial H}{\partial \alpha} = A(\lambda)P(t) \quad , \qquad P(0)=P_0 \quad ,$$

$$\dot{\alpha}(t) = -\frac{\partial H}{\partial P} = -A'(\lambda)\alpha(t) - B' \quad , \qquad \alpha(S)=0 \quad .$$

In addition, since for an optimal control $\lambda^*=(\lambda^*(Q_t),F_t)$, $0\leq t\leq S$, the Hamiltonian assumes its global, or absolute minimum, we have:

$$BP^*(t)+\alpha^{*\prime}(t)A(\lambda^*)P^*(t)\leq BP^*(t)+\alpha^{*\prime}(t)A(\lambda)P^*(t) \quad , \tag{7}$$

for all $\lambda\in\Lambda$, where $P^*(t)$ and $\alpha^*(t)$ are the state and costate trajectories corresponding to the optimal control λ^*.

From (7) we have:

$$\alpha^{*\prime}(t)[A(\lambda^*)-A(\lambda)]P^*(t)\leq 0 \quad ,$$

for all $\lambda\in\Lambda$. Thus

$$\sum_{k=0}^{N-1}[\lambda^*_k(t)-\lambda_k(t)][\alpha^*_{k+1}(t)-\alpha^*_k(t)]p^*_k(t)\leq 0 \quad , \tag{8}$$

for all $\lambda\in\Lambda$.

Since $p^*_k(t)\geq 0$, for all k, $0\leq k\leq N-1$, inequality (8) is satisfied if

$$[\lambda^*_k(t)-\lambda_k(t)][\alpha^*_{k+1}(t)-\alpha^*_k(t)]\leq 0 \quad .$$

for all k, $0\leq k\leq N-1$. The latter inequality holds if

$$\lambda^*_k(t) = \begin{cases} c & \alpha^*_{k+1}(t)-\alpha^*_k(t)<0 \\ \text{undetermined} & \alpha^*_{k+1}(t)-\alpha^*_k(t)=0 \\ 0 & \alpha^*_{k+1}(t)-\alpha^*_k(t)>0 \quad , \end{cases}$$

for all k, $0\leq k\leq N-1$.

Let us choose the integer L (window size) such that:

$$T^{(L-1)}_{\max}<T\leq T^{(L)}_{\max} \quad .$$

To obtain the optimal control of the M/M/1 queue let us first assume that the average system time delay is $T^{(L-1)}_{\max}$. In order to achieve the maximum throughput the optimum control $\lambda^*=(\lambda^*(Q_t),F_t)$, $0\leq t\leq S$, has to be chosen as follows:

$$\lambda^*_k(t) = \begin{cases} c & 0\leq k\leq L-2 \\ 0 & L-1\leq k<N \quad . \end{cases} \tag{9}$$

If, on the other hand, the average system time delay is $T^{(L)}_{\max}$ the optimal flow control policy is given by:

$$\lambda^*_k(t) = \begin{cases} c & 0 \le k \le L-1 \\ 0 & L \le k < N\ . \end{cases} \tag{10}$$

From (9) and (10) we conclude that the optimal control is given by:

$$\lambda^*_k(t) = \begin{cases} c & 0 \le k \le L-2 \\ \lambda_{L-1} & k = L-1 \\ 0 & L \le k < N\ . \end{cases} \qquad \square$$

IV. Conclusions

The dynamic optimal flow control of a M/M/1 queueing system that models the data link protocol in a computer communication network has been investigated. The throughput and the average system time delay of the M/M/1 queue over a finite time interval have been defined. The optimality criterion employed maximizes the throughput of the system under a maximum average system time delay constraint. Pontryagin's minimum principle has been employed to show that the optimum solution is a "bang-bang" control. The optimal flow control scheme is, therefore, a window flow control mechanism. The window size depends on the maximum tolerable time delay T, the maximum admissible load c, the duration of the time interval S and the initial number of packets in the system.

Acknowledgement

The research reported here was supported in part by the National Science Foundation under Grant # ECS-81-06476 and in part by the Office of Naval Research under Contract N00014-82-K-0508.

References

[1] Bremaud, P., "Bang-Bang Controls of Point Processes", *Advanced Applied Probability,* 8, 1976.

[2] Bremaud, P., *Point Processes and Queues : Martingale Dynamics,* Springer-Verlag, New York, 1981.

[3] Ephremides, A., Varaiya, P. and Warland, J., "A Simple Dynamic Routing Problem", *IEEE Trans. on Automatic Control,* Vol. **AC-25,** No. 4, August 1980.

[4] Kirk, D., *Optimal Control Theory An Introduction,* Prentice-Hall, Englewood Cliffs, 1970, pp. 227 -240.

[5] Lazar, A.A., "Optimal Control of M/M/1 Queue", *Proceedings of The Nineteenth Annual Allerton Conference on Communications, Control and Computing,* University of Illinois, Urbana, Oct 8-10, 1981.

[6] Lazar, A.A., "Centralized Optimal Control of a Jacksonian Networks", *Proceedings of the Sixteenth Conference on Information Sciences and Systems,* Princeton University, March 17-19, 1982.

[7] Lee, E.B. and Markus, L., *Foundation of Optimal Control Theory,* Wiley, New York, 1967.

[8] Pontryagin, L.S., Boltyankii, V.G., Gamkrelidze, R.V., and Mishchenko, E.F., *Mathematical Theory of Optimal Processes,* Intersciece, New York, 1962.

[9] Stern, T.E., "Approximation of Queue Dynamics and Their Application to Adaptive Routing in Computer Communication Networks", *National Telecommunication Conference,* New Orleans, 1978.

[10] Takacs, L., *Introduction to the Theory of Queues,* Oxford University Press, New York, 1962, pp. 12 - 22.

[11] Tanenbaum, A.S., *Computer Networks,* Prentice Hall, Englewood Cliffs, 1981, pp. 137-164.

COMPUTING RELIABILITY FOR A GENERALIZATION OF SERIES-PARALLEL NETWORKS

JOSEPH A. WALD and CHARLES J. COLBOURN
Computational Science, Univ. of Saskatchewan, Saskatoon, Canada.

ABSTRACT

Partial 2-trees are a generalization of series-parallel networks. We present a specific algorithm for computing the probability that a specified set of sites in a partial 2-tree is connected.

We model a network by a probabilistic graph in which each edge has an associated (statistically independent) probability of availability. This network model has been used before in many contexts [1]. A standard reliability problem on such networks is *terminal-pair connectedness*, the probability that a given pair of nodes s and t are connected. Another reliability problem is *probabilistic connectedness*, the probability that all the nodes are connected. A generalization of both problems is the *probabilistic Steiner tree* (PST) problem: given a probabilistic graph $G=(V,E)$ and a target set of nodes T, what is the probability PST(G,T) that all nodes in T are connected? This generalizes terminal-pair connectedness (where $T=\{s,t\}$) and probabilistic connectedness (where $T=V$).

All known exact methods to solve these reliability problems require exponential time in the worst case for arbitrary networks; this is expected in light of Valiant's result that the terminal-pair and probabilistic connectedness problems are #P-complete [4]. This has led researchers to consider restricted classes of networks -- in particular, series-parallel.

A *series-parallel* (S-P) graph can be recursively defined as follows: an edge $\{x,y\}$ is a S-P graph; given an edge $\{x,y\}$ of an S-P graph, adding node z adjacent to x and y yields an S-P graph whether or not edge $\{x,y\}$ is deleted. In the case of S-P graphs there is a polynomial time solution for the terminal-pair connectedness problem [3].

We consider a generalization of S-P networks here. A *2-tree* can be recursively defined as follows: an edge $\{x,y\}$ is a 2-tree; given an edge $\{x,y\}$ of an 2-tree, adding node z adjacent to x and y yields an 2-tree. A *partial 2-tree* is simply a subgraph of a 2-tree. It is easily seen that an S-P graph is a partial 2-tree and can be completed to a 2-tree by adding those edges which were deleted in the recursive construction.

A linear time algorithm for the PST problem on partial 2-trees has been formally developed in [5]. Since S-P networks are partial 2-trees, this provides a substantially more general method than the usual S-P reliability techniques. Our PST algorithm for partial 2-tree works in two phases. In the first phase we add edges to a partial 2-tree to construct a 2-tree. These edges are "virtual" in the sense that they have probability zero of availability. In the second phase we recursively delete edges from the 2-tree while summarizing relevant information in the remaining edges.

Consider the problem of adding edges to a graph to construct a 2-tree. We describe the major points here. See [2] for graph-theoretic terminology. First we transform an arbitrary graph to a connected graph. Next we transform the connected graph to a biconnected graph. Finally, we transform the biconnected graph G to a 2-tree (or we discover that the original graph was not a partial 2-tree). The final step is achieved as follows:

```
Form a set containing all nodes of degree 2.
Repeat until the set is empty:
    Remove node v from the set and locate neighbours x and y of v.
    If {x,y} is not an edge of G, then add it to G. Delete vertex v.
    If x or y now has degree 2, then add it to the set.
```

If G is not an edge when the set is empty, then the original graph was not a partial 2-tree. Otherwise, we have completed the graph in linear time.

The second phase involves solving PST for 2-trees. Our strategy is to reverse the recursive construction process of the 2-tree -- that is, repeatedly eliminate vertices of degree two until the graph becomes a single edge. We are given a 2-tree G with nonempty target set T; every edge e of G has an associated probability RELY[e]. With each arc (x,y) of G, we associate six probability measures, which summarize the probabilities in the subgraph S which has so far been reduced onto the edge {x,y}:

1. sc(x,y) is the probability that a target-connected subgraph (TCS) of S
contains both x and y. 2. dc(x,y) is the probability that all the
targets in subgraph S are connected to x or y but not both.
3. yn(x,y) is the probability that a TCS of S contains x but not y.
4. ny(x,y) is the probability that a TCS of S contains y but not x.
5. nn(x,y) is the probability that a TCS of S contains neither x nor y.
6. no(x,y) is the probability that S contains no targets.

We set the initial measures of probabilities as follows:

1. sc(x,y)= RELY[e]. 2. dc(x,y)= 1-RELY[e].
3. yn(x,y)= if $y \in T$ then 0; else 1-RELY[e].
4. ny(x,y)= if $x \in T$ then 0; else 1-RELY[e].
5. nn(x,y)= 0. 6. no(x,y)= if $x \in T$ or $y \in T$ then 0; else 1.

These initial arc probabilities are used to recalculate arc probabilities as the graph is reduced. Suppose at some point there is a triangle {A,B,C} in which B is a degree two vertex. Probabilities have been computed for arcs L=(A,B) and R=(B,C); we use these to recalculate the probabilities for arc M=(A,C) using six recurrences:

temp= sc(L)*dc(R) + dc(L)*sc(R) + yn(L)*ny(R).
1. sc(M)= sc(M)*(sc(L)*sc(R) + temp) + dc(M)*sc(L)*sc(R).
2. dc(M)= dc(M)*temp. 3. yn(M)= yn(M)*(sc(L)*yn(R) + yn(L)*no(R)).
4. ny(M)= ny(M)*(ny(L)*sc(R) + no(L)*ny(R)).
5. nn(M)= nn(M)*no(L)*no(R) + no(M)*(nn(L)*no(R)+no(L)*nn(R)+ny(L)*yn(R)).
6. no(M)= no(M)*no(L)*no(R).

These measures are computed symmetrically for arc (C,A) after which B is deleted. Repeating this process, ultimately G is a single edge {x,y}. At this point, the overall probability of obtaining a subgraph connecting all target vertices, PST(G,T) is taken to be sc(x,y)+yn(x,y)+ny(x,y)+nn(x,y). This completes the algorithm. See [5] for proof of timing and correctness.

This two phase algorithm has been improved to enhance its practical efficiency by merging the two phases so that virtual edges are added only as they are needed [6]. The linear time of our algorithm guarantees better performance than non-linear algorithms on sufficiently large problems. We have also found that our algorithm remains competitive on small problems where there already exist efficient solutions.

REFERENCES

[1] H. Frank and I.T. Frisch, Communications, Transmission, and Transportation Networks, Addison-Wesley, 1971.
[2] F. Harary, Graph Theory, Addison-Wesley, Reading MA, 1972.
[3] F. Moskowitz, "The analysis of redundancy networks", AIEE Trans. Commun. Electron. 39 (1958), 627-632.
[4] L.G. Valiant, "The complexity of enumeration and reliability problems", SIAM J. Comput. 8 (1979), 410-421.
[5] J.A. Wald and C.J. Colbourn, "Steiner trees in probabilistic networks", Tech. Report 82-7, Dept. of Comp. Science, Univ. of Sask., 1982.
[6] J.A. Wald and C.J. Colbourn, "Algorithm Prototype for Partial 2-Trees", Tech. Report to appear, Dept. of Comp. Science, Univ. of Sask., 1982.

AN AVERAGE MESSAGE DELAY CONSTRAINED MINIMUM HOP FLOW ASSIGNMENT AND ROUTING ALGORITHM[†]

J. S. MEDITCH
Department of Electrical Engineering
University of Washington
Seattle, WA 98195

M. S. CHEN[††]
Department of Electrical Engineering
and Computer Science
Polytechnic Institute of New York
Brooklyn, NY 11201

ABSTRACT

A novel approach to the problem of minimum hop flow assignment and routing in message-switched networks subject to an average message delay constraint is developed and illustrated. The principal result is a centralized algorithm for performance analysis studies. The approach taken in developing the algorithm is based on a mapping of the feasible region of solutions onto the "delay vs. hop" plane. This mapping then permits one to solve the problem via successive utilization of the flow deviation method. The algorithm is such that it obtains the desired solution directly if one exists or shows that no solution exists. The latter occurs when there is no feasible flow assignment which can satisfy the delay constraint. In this case, the algorithm can be used to generate the performance curve of minimum achievable delay vs. number of hops for the network. The algorithm is illustrated via application to a 10-node, 36-link network with 40 commodities.

1. INTRODUCTION

In the classical formulation of the flow assignment and routing problem for message-switched networks, the performance measure is that of average message delay over all source-destination node pairs. More recent work [1-3] has focused on minimum hop flow assignment and routing subject to a set of one or more end-to-end average delay constraints. The effect of minimum hop routing is to minimize the number of times that messages must be stored-and-forwarded by intermediate modes. This tends to reduce nodal processing delay and network control traffic. The inclusion of end-to-end delay constraints then insures timely delivery of messages to their respective destination nodes. An extension of the research on minimum hop routing is that of replacing the set of end-to-end delay constraints by a single average message delay constraint over all source-destination node pairs. This is more in keeping with the traditional formulation of the flow assignment and routing problem, and provides an alternative to end-to-end delay constraints which may be more useful in certain applications.

In this paper, we develop and illustrate a centralized minimum hop routing algorithm subject to an overall average message delay constraint.

[†]Research sponsored in part by the National Science Foundation under NSF Grant ECS-8011262 and in part by the U.S. Air Force Office of Scientific Research under AFOSR Grant 78-3546.

[††]Formerly with Department of Electrical Engineering, University of Washington, Seattle, WA 98195

Our presentation is as follows. We formulate the problem in Section 2, and develop an algorithm which solves it in Section 3. We illustrate the algorithm in Section 4 via an example of a 10-node, 36-link network with 40 commodities, and conclude in Section 5 with a discussion of our results and an indication of some directions for future research.

2. PROBLEM FORMULATION

We consider a message-switched network with N nodes and M directed links in which message lengths are exponentially distributed with mean $1/\mu$ bits, and the network message arrival processes are independent and Poisson. We let γ^{sd} be the average rate in messages/sec at which traffic enters the network at a source node s with some other node d as its destination, and let Γ_γ denote the set of all source-destination node pairs (s,d). Each $(s,d) \in \Gamma_\gamma$ then specifies a distinct commodity for the network. We define C_{ij} to be the capacity of link (i,j) in bits/sec, and let λ_{ij} be the total average flow on that link in messages/sec, $i, j = 1, \ldots, N$, $i \neq j$. Also, we let Γ_C denote the set of all links in the network. Finally, we assume that the message arrival and transmission times at each node are independent of each other. The basic network model here is then the one due to Kleinrock [4].

Generally, we shall use a pair of superscripts to denote respective source-destination nodes, and a pair of subscripts to represent the nodes of a link. Hence, λ_{mn}^{sd} specifies the average flow on the directed link (m,n) where that flow originates at node s and has node d as its destination. Since the flow assignment problem is one of specifying paths for multicommodity flows, we let Π_ℓ^{sd} be the set of all links which comprise the ℓth shortest path for the commodity (s,d) and let Π^{sd} denote the union of all such paths. Here, path length is defined to be the number of links in a loop-free directed path from node s to node d. Hence, Π_1^{sd} denotes the path with the fewest links, Π_2^{sd} the one with the next fewest links, and so forth. Where two or more (s,d) paths have the same length, they are indexed separately. Finally, paths need not be disjoint, i.e., two or more (s,d) paths may share one or more links among themselves as well as with other source-destination paths.

After determining paths for each commodity (s,d), we need to specify how much of the average (s,d) traffic will utilize each of these paths. We define the routing variable r_ℓ^{sd} to be the average fraction of γ^{sd} which will flow over the path Π_ℓ^{sd}. This formulation particularly facilitates satisfying the constraints of nonnegativity of flow and conservation of commodity flow. In particular, we have, for every $(s,d) \in \Gamma_\gamma$, that

$$r_\ell^{sd} \geq 0 \tag{1}$$

and

$$\sum_\ell r_\ell^{sd} = 1 \tag{2}$$

respectively.

Upon specifying the paths and their respective routing variables, we are able to evaluate the average number of hops over all commodities. This is given [1-4] by

$$J = \frac{1}{\gamma} \sum_{(m,n)\in\Gamma_C} \lambda_{mn} \tag{3}$$

where γ is the total average input traffic,

$$\gamma = \sum_{(s,d)\in\Gamma_\gamma} \gamma^{sd} \tag{4}$$

and λ_{mn} is subject to the link capacity constraints

$$\lambda_{mn} \leq \alpha_{mn}\mu C_{mn} \tag{5}$$

where $0 < \alpha_{mn} < 1$ and $(m,n) \in \Gamma_C$. The use of α_{mn} in Eq. (5) has the effect of limiting the average utilization $\rho_{mn} = \lambda_{mn}/\mu C_{mn}$ to a maximum of α_{mn} for each link, a feature which is desirable in practice.

To specify the delay constraint, we let D be the average message delay for the entire network and require it to be less than or equal to a prespecified value D_{max}. From [4], D is given by the equation

$$D = \frac{1}{\gamma} \sum_{(m,n)\in\Gamma_C} \left[\frac{\lambda_{mn}}{\mu C_{mn} - \lambda_{mn}} + P_{mn} \lambda_{mn} + K \lambda_{mn} \right] \tag{6}$$

where K is the nodal processing delay under the assumption that all nodes have the same processing delay, P_{mn} is the propagation delay over link (m,n), and the first term inside the bracket is the average queueing and service delay for link (m,n). In many studies, P_{mn} and K are omitted since they are relatively small compared to the queueing and service delay. However, P_{mn} is not negligible when satellite links are present. Here, we include the latter, but assume that K is negligible. The delay constraint we have is then

$$D = \frac{1}{\gamma} \sum_{(m,n)\in\Gamma_C} \left[\frac{\lambda_{mn}}{\mu C_{mn} - \lambda_{mn}} + P_{mn} \lambda_{mn} \right] \leq D_{max} \tag{7}$$

In summary, our problem is to minimize the average number of hops, Eq. (3), with respect to the set of routing variables $\{r_\ell^{sd}\}$ where the minimization is subject to nonnegativity of flow, Eq. (1), conservation of commodity flow, Eq. (2), link capacity constraints, Eq. (5), and the average message delay constraint, Eq. (7).

3. ALGORITHM

In the average message delay constrained minimum hop problem, we see that the objective function and constraints are linear except for the delay constraint which is convex in the λ_{mn}. Furthermore, the feasible solution set is a convex polyhedron, and the problem is, therefore, one of convex minimization.

Although there are numerous methods for convex minimization, the flow deviation (FD) method [4] is particularly attractive for our problem. This is the case since it is effective for problems with a convex objective function and linear constraints. While our problem has a linear objective function with convex constraints, we shall show how the FD method can be incorporated into an algorithm which solves the problem.

To proceed, we consider first the relationship between the average delay D and the average number of hops J. If we map the feasible solution set onto a plane which has the average number of hops J as the abscissa and the average delay D as the ordinate, we have the "D-J feasible" region in the D-J plane. Generally [5], this has the shape shown in Fig. 1. In the figure, A represents a flow assignment* which has average delay and average number of hops D_A and J_A, respectively. Similarly, B represents a flow assignment which has D_B and J_A as its respective average delay and average number of hops where the delay has been minimized. Therefore, if we start

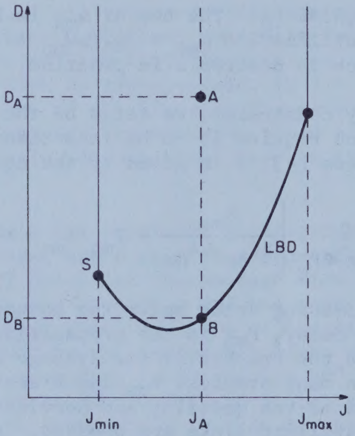


Fig. 1. Feasible region in D-J plane.

from an assignment represented by A and introduce the constraint $J = J_A$ on the average number of hops, and then use the FD method to minimize the average message delay with this constraint present in addition to the others, the result will be that represented by B. The same is true for every J as long as $J_{min} \le J \le J_{max}$ where J_{min} is the solution of the minimum (maximum) hop flow assignment problem, respectively, in the absence of the average message delay constraint. Hence, there is a lower bound on the delay for the feasible region which we denoted by LBD.

Now, since we are able to reach any point on the curve LBD, we need only discuss the relationship between LBD and the prespecified D_{max} as indicated in Fig. 2. In part (a) of the figure, D_{max} is greater than or equal to D_S and the solution of our problem is the one represented by the point S. If the previous condition is not true and the line D_{max} has one or more intersection(s) with LBD as shown in part (b), we choose the intersection which is closer to the point S, i.e., the one which minimizes J while satisfying $D \leq D_{max}$. If there is a point of tangency, then the solution is given by that point D_{min} in part (c) of the figure which is also the minimum delay assignment. In part (c), we also show the case where D_{max} is less than D_{min}. In this instance, the problem has no solution since there exists no flow assignment which can satisfy the delay constraint.

---

*Point A corresponds to a feasible network flow, i.e., one which uses allowable source-destination paths, and satisfies the nonnegativity of flow, conservation of flow, and capacity constraints, without regard to minimization of J or satisfaction of the average message delay constraint. Evaluation of J and D for this flow then results in the point $A = (J_A, D_A)$.

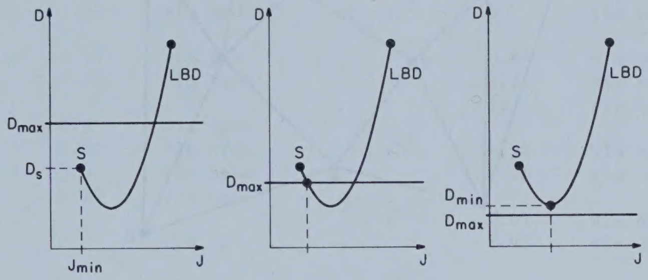


Fig. 2. Relationships between LBD and D_{max}.

From the above arguments, it is clear that the desired solution is the point on LBD which is closest to S and has delay less than or equal to D_{max}. Otherwise, there is no solution at all. Therefore, in our algorithm, we shall always start from point S and move to the right if necessary. Any convenient one-dimensional search technique is applicable. We have used a direct search in which we step away from J_{min} in increments of ΔJ.

The centralized overall average delay constrained minimum hop flow assignment and routing algorithm which we thus have now follows.

Step 1. Solve for J_{min} and J_{max}[**], and set the stepsize parameter ΔJ.

Step 2. Set J to J_{min} and D_{min} to a large number.

Step 3. Determine the point on LBD using the FD method subject to the constraint on the average number of hops J and let D denote the corresponding average delay.

Step 4. If $D \leq D_{max}$, stop. The solution is found. Otherwise, go to the next step.

Step 5. If $D > D_{min}$ or $J + \Delta J > J_{max}$, stop. There is no solution. Otherwise, let $D_{min} = D$ and $J = J + \Delta J$ and go to Step 3.

4. EXAMPLE

We illustrate the above algorithm by applying it to the network shown in Fig. 3 where the nodes are labeled a through j with node j denoting a geosynchronous satellite. All links are assumed to be full duplex with capacity $C_{mn} = C_{nm}$. The capacity of each terrestrial link is taken to be 38.4 KBPS and that of each satellite link is 50 KBPS. Each satellite link is assumed to introduce a propagation delay of 125 msec.

[**] This is done via a linear program.

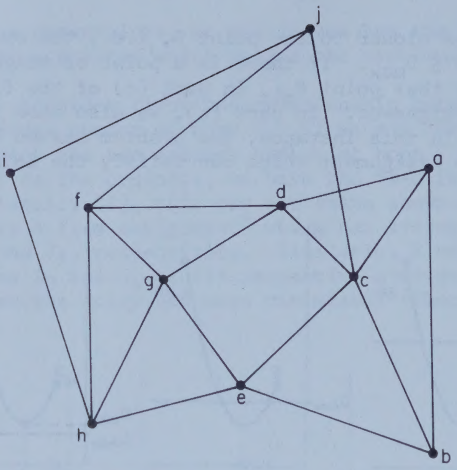


Fig. 3. Network for flow assignment example.

The average message length is assumed to be 1000 bits. For the source-destination node pairs (b,f), (c,f), and (h,f), we take $\gamma^{sd} = 3$; for (a,b), (a,c), (a,f), (b,a), (b,d), (b,e), (c,a), (c,b), (c,d), (c,g), (c,h), (d,a), (d,e), (d,g), (d,h), (d,i), (e,a), (e,f), (e,i), (f,b), (f,c), (f,e), (f,i), (g,a), (g,b), (g,f), (h,d), (h,e), and (i,e), $\gamma^{sd} = 5$; for (a,h), (b,c), (b,h), (h,c), and (i,h), $\gamma^{sd} = 10$; for (h,a), $\gamma^{sd} = 15$; and for (c,i) and (i,c), $\gamma^{sd} = 20$ where the units are messages/sec. This 10 node network thus has 36 directed links and 40 distinct commodities.

The problem which we address is that of obtaining a minimum hop flow assignment which has an average message delay less than or equal to 0.1700 sec to within a tolerance of 0.05 on J ($\Delta J=0.05$).

The first step in solving this problem is to determine the paths and then define the routing variables. In this example, we use all source-destination paths which have no more than 4-hops [5]. For the 40 commodities of input traffic, there are 433 paths available, and J_{min} is 1.8456. After performing Step 3 of the algorithm with the constraint on the average number of hops set to 1.8456, the minimum delay obtained is 0.2424 sec. Since the delay constraint is not satisfied, we increase J by ΔJ to 1.8956 and repeat Step 3 of the algorithm. The minimum delay obtained in this iteration is 0.1748 which is still greater than D_{max}. We then increase J to 0.19456 and perform Step 3 again. This time, we reach an assignment in which the delay is 0.1643 sec, and the solution is found. We list the complete set of corresponding commodity flow assignments in Table 1.

5. CONCLUSION

In this paper, we have developed and illustrated a centralized minimum hop flow assignment and routing algorithm for message-switched networks subject to an overall average message delay constraint. The use of a mapping of the feasible region of solutions onto the D-J plane led to a simple and efficient procedure for solving the problem via successive utilization of the flow deviation method. The results were illustrated via

γ^{ba}:	5	via ba
γ^{ca}:	5	via ca
γ^{da}:	5	via da
γ^{ea}:	3.36	via eca
	0.57	via eba
	1.07	via egda
γ^{ga}:	3.17	via gda
	1.83	via geba
γ^{ha}:	6.3	via hgda
	3.22	via hfda
	5.32	via heba
	0.16	via hgeba
γ^{ab}:	5	via ab
γ^{cb}:	5	via cb
γ^{fb}:	1.83	via fgeb
	1.07	via fjcb
	2.1	via fdab
γ^{gb}:	3.93	via geb
	1.07	via gdcb
γ^{ac}:	5	via ac
γ^{bc}:	10	via bc
γ^{fc}:	1.07	via fjc
	3.93	via fdc
γ^{hc}:	5.8	via hec
	4.2	via hgdc
γ^{ic}:	20	via ijc
γ^{bd}:	3.17	via bcd
	1.83	via bad
γ^{cd}:	5	via cd
γ^{hd}:	3.74	via hgd
	1.26	via hfd
γ^{be}:	5	via be
γ^{de}:	1.07	via dge
	3.93	via dce
γ^{fe}:	1.07	via fhe
	3.93	via fge
γ^{he}:	2.9	via he
	2.1	via hge
γ^{ie}:	5	via ihe
γ^{af}:	5	via adf
γ^{bf}:	0.64	via behf
	1.26	via begf
	1.10	via badf
γ^{cf}:	0.64	via cjf
	2.36	via cdf
γ^{ef}:	1.07	via ehf
	3.43	via egf
γ^{gf}:	4.24	via gf
	.76	via ghf
γ^{hf}:	1.9	via hf
	1.1	via hgf
γ^{cg}:	3.08	via ceg
	1.92	via cdg
γ^{dg}:	5	via dg
γ^{ah}:	5.05	via adgih
	3.65	via adfih
	1.3	via acjih
γ^{bh}:	9.61	via beh
	0.39	via begh
γ^{ch}:	2.9	via ceh
	2.1	via cdgh
γ^{dh}:	3.17	via dgh
	1.83	via dfh
γ^{ih}:	10	via ih
γ^{ci}:	15.43	via cji
	4.57	via cehi
γ^{di}:	1.83	via dfhi
	2.1	via dghi
	1.07	via dacji
γ^{ei}:	2.9	via ehi
	2.1	via eghi
γ^{fi}:	1.07	via fji
	3.93	via fhi

Table 1. Optimal flow assignment for network example.

application to a network with 10 nodes, 36 links, and 40 commodities.

Since the algorithm is a centralized one, its principal value is for performance analysis studies. In particular, we note its use for generating the network's performance curve (LBD in Fig. 1) of minimum achievable delay vs. number of hops between the two extremes of flow assignment corresponding to J_{min} and J_{max}, respectively.

A logical next step in this research is that of developing nodewise distributed routing algorithms for the problem of interest here. This is currently being examined from the viewpoint of distributed convex programming. The primary issue, however, is that of whether or not the global problem is nodewise distributable subject to reasonable assumptions on the amount of information exchange to be permitted among the nodes.

REFERENCES

1. J. S. Meditch and F. D. Gorecki, "Minimum hop flow assignment and routing in computer-communication networks," Proc. 19th IEEE Conf. on Decision and Contr., 1980, pp. 634-636. (Invited paper.)

2. ————, "A distributed minimum hop routing algorithm, Proc. 20th IEEE Conf. on Decision and Contr., 1981, pp. 392-297.

3. F. D. Gorecki and J. S. Meditch, "On minimum hop flow assignment in message-switched telecommunication networks," Proc. 8th IFAC Congress, 1981, vol. XI, pp. 148-153.

4. L. Kleinrock, Queueing Systems, Volume II: Computer Applications. New York: Wiley-Interscience, 1976.

5. M. S. Chen, Routing Algorithms in Message-Switched Computer Networks, M. S. thesis, Dept. of Elec. Engr., Univ. of Wash., Seattle, Wash., May 1982. (Also Tech. Rept. No. 223.)

THE ULTIMATE PLANAR CONVEX HULL ALGORITHM?[1]

DAVID G. KIRKPATRICK and RAIMUND SEIDEL
Dept. of Computer Science
Univ. of British Columbia
Vancouver, B.C., Canada V6T 1W5

Dept. of Computer Science
Cornell University
Ithaca, NY 14853

ABSTRACT

The construction of the convex hull of finite planar point sets has been one of the most extensively studied problems in computational geometry. Several algorithms for this problem have been proposed so far. Some of them have O(nlogn) worst case time complexity and have been shown to be optimal in a certain sense. For others the time bound is O(nH), i.e. it also depends on H, the number of edges found to be on the hull. Thus the choice of the best algorithm appears to depend on the size of H which is not usually known in advance. We abolish this dilemma by presenting an algorithm with worst case time bound O(nlogH).

1. INTRODUCTION

The convex hull of a finite point set S in the plane is the smallest convex polygon containing the set. The vertices (corners) of this polygon must be points of S. Thus in order to compute the convex hull of a set S it is necessary to find those points of S which are vertices of the hull. The convex hull problem, as we define it, is to construct the ordered sequence of points of S which constitute the sequence of vertices around the convex hull.

The convex hull problem was one of the first problems in the field of computational geometry to have been studied from the point of view of computational complexity. In fact, efficient algorithmic solutions were proposed even before the term "computational geometry" was coined. This, along with its very extensive analysis in recent years, reflects both the theoretical and practical importance of the problem.

We restrict our attention to the worst case complexity of the convex hull problem. Of the algorithms proposed so far several have O(nlogn) worst case time bounds [6,9,10,11], where n is the size of the input point set. Shamos [11] even argued that the O(nlogn) time bound is worst case optimal. He observed that a set S of n real numbers could be sorted by finding the convex hull of the planar set $S' = \{(x,x^2) | x \text{ in } S\}$. But sorting, of course, has an Ω(nlogn) lower bound on a wide range of computational models. Preparata and Hong [10] proved the same lower bound by reduction to the maximal vector problem which was shown to be Ω(nlogn) by Kung et al. [8]. Yao [12] and to a lesser extent van Emde Boas [5] and Avis [2] also proved the Ω(nlogn) bound for a less demanding version of the convex hull problem: just the vertices of the convex hull are to be identified, irrespective of their sequence.

In contrast to the results above, it is interesting to observe that algorithms exist which solve the planar convex hull problem in O(nH) time, where H is the number of edges found to be on the hull [4,7]. For small H these algorithms seem to be superior to the O(nlogn) methods. (This, of

---

[1] This research was supported by the Natural Science and Engineering Research Council of Canada under Grant A3583.

course, does not contradict the previously cited lower bound results, as H could also be as large as n). It is notable that all of the lower bound arguments mentioned above are insensitive to H in that they assume that some fixed fraction of the data points are vertices of the convex hull.

This shortcoming of the lower bound results is further emphasized by the convex hull algorithm of this paper. It has an O(nlogH) worst case time bound and is thus not only superior to the O(nH) methods even for small H, but also worst case optimal in the traditional sense. The crux of our method is to identify in linear time a previously unknown edge of the hull and simultaneously to split the problem in two halves. The algorithm is described in detail in sections 2 and 3.

Throughout this paper we will deal with sets of points in the plane. For a point p, x(p) and y(p) denote its standard cartesian coordinates. We will feel free to use loose but descriptive geometric terminology such as "vertical line", "a point lies above a line", etc.

2. THE MAIN ALGORITHM

In this section we describe our new convex hull algorithm and prove its O(nlogH) worst case time bound. However, the algorithm and its complexity rely heavily on a linear time solution for, what we call, the "bridge problem" which we present in the next section.

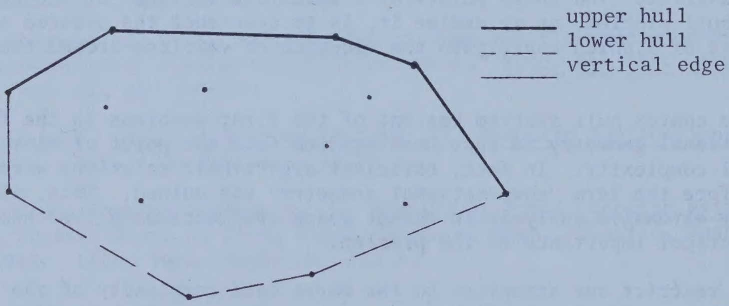


Figure 2.1

We construct the convex hull in two pieces, the upper hull and the lower hull (see Figure 2.1). It should be clear that if the two chains forming the upper and lower hull are given, they can be concatenated in constant time (at most two vertical edges may need to be inserted) to yield the sequence of vertices around the hull. Also observe that an algorithm for constructing the upper hull could easily be modified to construct the lower hull also. Finally observe that if such an algorithm runs in $O(n\log H_U)$ time, then the entire convex hull can be constructed in $O(n\log H_U + n\log H_L) = O(2n\log H) = O(n\log H)$ worst case time (H_U and H_L are the number of edges on the upper and lower hull respectively). Consequently it suffices to concentrate on an $O(n\log H_U)$ algorithm to determine the sequence of vertices on the upper hull.

The basic idea of our algorithm for this problem is quite simple. We first find a vertical line which divides the given point set in two approximately equal sized parts and determine the "bridge" crossing this line, i.e. the unique edge of the upper hull which intersects this line. We then apply this algorithm recursively to point sets on the left and right side of the vertical line.

The following PIDGIN-ALGOL routine presents the algorithm in more detail. It takes as input a set $S = \{p_1,\dots,p_n\}$ of n points in the plane and prints out the sequence of indices of the vertices on the upper hull of S. It is assumed that a function BRIDGE(S,A) is known which returns a pair (i,j) such that p_i and p_j are the left and right endpoints of the unique edge (if any) of the upper hull of S which crosses the vertical line $\{(x,y) \mid x=A\}$. Such a function is given in the next section.

Procedure UPPER HULL(S)

```
1.   Let min and max be the indices of the left and right endpoints of the
     upper hull respectively, i.e.
         $x(p_{min}) \le x(p_i) \le x(p_{max})$ and
         $y(p_{min}) \ge y(p_i)$ if $x(p_{min}) = x(p_i)$
         $y(p_{max}) \ge y(p_i)$ if $x(p_{max}) = x(p_i)$
              for i=1,...,n.

2.   CONNECT(min,max,S)
       where CONNECT(k,m,S) is
       BEGIN
2.1      Find a real number A such that
             $x(p_i) < A$ for $\lceil |S|/2 \rceil$ points in S and
             $x(p_i) > A$ for $\lfloor |S|/2 \rfloor$ points in S.

2.2      Find the "bridge" over the vertical line $L = \{(x,y) \mid x=A\}$, i.e.
             (i,j) := BRIDGE(S,A)

2.3      Let $S_{left} = \{p \text{ in } S \mid x(p) \le x(p_i)\}$.
         Let $S_{right} = \{p \text{ in } S \mid x(p) \ge x(p_j)\}$.

2.4      If i=k then print(i)
                else CONNECT(k,i,$S_{left}$).
         If j=m then print(j)
                else CONNECT(j,m,$S_{right}$).
       END.
```

Theorem 2.1

The above algorithm correctly determines the sequence of vertices on the upper hull of S in O(n) space and $O(n \log H_U)$ time, where H_U is the number of vertices on the upper hull.

Proof

The correctness of the algorithm follows from an inductive argument. A call CONNECT(k,m,S) discovers a previously unknown edge (p_i,p_j) on the upper hull of S. If p_i turns out to be the leftmost vertex of the upper hull its index will be printed, otherwise the recursive call CONNECT(k,i,S_{left}) will cause the sequence of vertices of the upper hull from p_k up to p_i to be printed. Similarly, if p_j is the rightmost vertex of the upper hull its index will be printed, otherwise the call CONNECT(j,m,S_{right}) will cause the portion of the upper hull from p_j up to p_m to be printed.

For the complexity bounds first observe that step 1 of the algorithm can be easily implemented in linear time. Thus it remains to show that the procedure CONNECT takes no more than $O(n \log H_U)$ time. Note that the procedure without the recursive calls needs $O(|S|)$ time: This is trivial for step 2.3, step 2.1 can be done in $O(|S|)$ time by a slight modification of the median algorithm by Blum et al. [1, pp.97-99] (the real A must not be equal to the x-coordinate of any point in S), and step 2.2 can be performed in linear time by Theorem 3.1. Thus the overall time needed by this procedure is bounded by $f(n,H_U)$, where

$$f(n,h)=\begin{cases} cn & h=2 \\ cn + \max\limits_{h_1+h_2=h} \left(f(\frac{n}{2},h_1) + f(\frac{n}{2},h_2)\right) & h>2,\ c \text{ some constant.}\end{cases}$$

Using the fact that $\max\limits_{h_1+h_2=h} (\log(h_1 * h_2)) = \log\frac{h^2}{4}$, it is straightforward to show that $f(n,h)$ is bounded from above by $bn\log h$ for any constant $b>c/2$ which completes the proof of the time bound.
The linear space bound is trivial.

Q.E.D.

3. FINDING THE BRIDGE

This section deals with the bridge problem which we define as follows: Given a set S of n points in the plane and a vertical line L which has points of S to its left and right and contains no point of S, determine the unique edge of the upper hull of S which intersects L. We call this unique edge the bridge and its endpoints bridge points (see Figure 3.1).

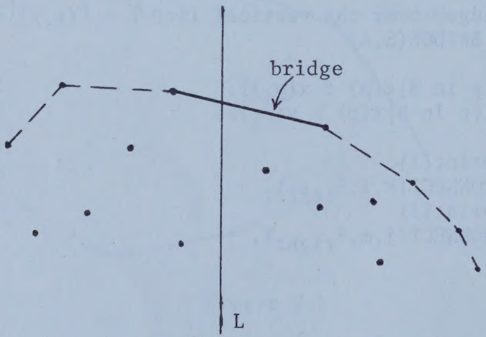


Figure 3.1

For the purposes of this section we define a supporting line of S to be a straight non-vertical line which contains at least one point of S but has no points of S above it.

We will give an $O(n)$ algorithm for the bridge problem which uses the strategy of recursively discarding a fixed fraction of points of S, all of which could not possibly be bridge points. The following lemma justifies the discarding policy to be used in our algorithm.

Lemma 3.1

Let S and L be as above. Let p and q be two points in S with $x(p) \le x(q)$, and let g be the straight line passing through p and q. Finally, let $h = \{(x,y) | y=kx+d\}$ be a supporting line of S with slope k.

1) If $x(p)=x(q)$, then the point with the smaller y-coordinate cannot be a bridge point.

Otherwise:

2) If h contains points of S both to the left and right of L, then h must contain the bridge.
3) If h contains no points of S to the right of L and if the slope of g is greater than or equal to k, then p cannot be a bridge point.
4) If h contains no points of S to the left of L and if the slope of g is smaller than or equal to k, then q cannot be a bridge point.

Proof

1) and 2) are trivial.
3) Assume that p was a bridgepoint.
As h is a supporting line of S which contains only points of S to the left of L, the supporting line b which contains the bridge must have a smaller slope than h. As, by assumption, the slope of g is greater than or equal to the slope of h, it must be greater than the slope of b. But this implies that if p lies on b, then q lies above b, which is a contradiction as no point of S can lie above a supporting line (see Figure 3.2).

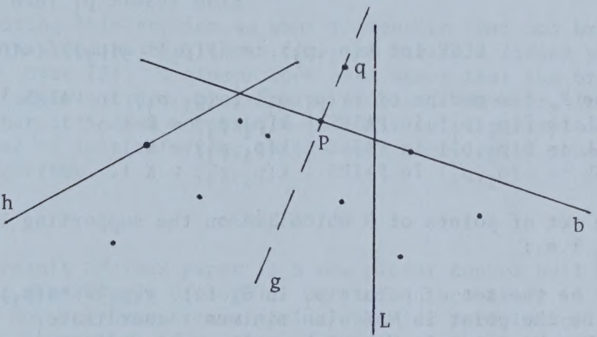


Figure 3.2

4) Symmetric to 3).

Q.E.D.

With this result it is easy to see how a fixed fraction of the points can be eliminated in linear time. Pair up the points of S into couples defining $\lfloor n/2 \rfloor$ straight lines. Find the median K of the slopes of these lines, and determine the supporting line of S with slope K. If this supporting line contains points of S on both sides of L, then it also contains the bridge. If it contains no points of S to the right of L, then a quarter of the points of S can be discarded using point 3) of Lemma 3.1; if it contains no point of S to the left of L, then a quarter of the points can be eliminated using point 4) of this lemma. The following algorithm gives more detail.

The function BRIDGE(S,A) takes as parameters a set $S = \{p_1,\ldots,p_n\}$ of $n>1$ points and a real A representing the vertical line $L = \{(x,y) \mid x=A\}$. It is assumed that there are points of S both to the left and right of L, but no points on L. BRIDGE(S,A) returns as its value a pair (i,j), where p_i and p_j are the left and right bridge point respectively.

```
Function  BRIDGE(S,A)

0.  Let CANDIDATES be empty.

1.  If $|S|<4$ then find the left and right bridgepoint $p_i$ and $p_j$ by
    some brute force method and RETURN( (i,j) ).

2.  Choose $\lfloor |S|/2 \rfloor$ disjoint sets of size 2 from S.
    If a point of S remains, then insert it into CANDIDATES.
    Arrange each subset to an ordered pair $(p_i,p_j)$, s.t. $x(p_i)<x(p_j)$.
    Let PAIRS be the set of these pairs.

3.  Determine the slopes of the straight lines defined by the pairs.
    In case the slops does not exist for some pair, apply point 1) of
    Lemma 3.1, i.e.:

      FOR all $(p_i,p_j)$ in PAIRS DO
        IF $x(p_i)=x(p_j)$ THEN delete $(p_i,p_j)$ from PAIRS
                                IF $y(p_i)>y(p_j)$ THEN insert $p_i$ into CANDIDATES
                                                   ELSE insert $p_j$ into CANDIDATES

                           ELSE let $k(p_i,p_j) := (y(p_i) - y(p_j))/(x(p_i)-x(p_j))$.

4.  Determine K, the median of $\{k(p_i,p_j) \mid (p_i,p_j)$ in PAIRS $\}$.
    Let SMALL := $\{(p_i,p_j)$ in PAIRS $\mid k(p_i,p_j) < K \}$.
    Let EQUAL := $\{(p_i,p_j)$ in PAIRS $\mid k(p_i,p_j) = K \}$.
    Let LARGE := $\{(p_i,p_j)$ in PAIRS $\mid k(p_i,p_j) > K \}$.

5.  Find the set of points of S which lie on the supporting line h with
    slope K, i.e.:

      Let MAX be the set of points $p_i$ in S, s.t. $y(p_i)-K*x(p_i)$ is maximal.
      Let $p_k$ be the point in MAX with minimum x-coordinate.
      Let $p_m$ be the point in MAX with maximum x-coordinate.

6.  Determine if H contains the bridge, i.e.:

      IF $x(p_k) < A$ and $x(p_m) > A$ THEN  RETURN( (k,m) ).

7.  h contains only points to the left of L:

      IF $x(p_m) < A$ THEN
             FOR all $(p_i,p_j)$ in LARGE $\cup$ EQUAL insert $p_j$ into CANDIDATES.
             FOR all $(p_i,p_j)$ in SMALL insert $p_i$ and $p_j$ into CANDIDATES.

8.  h contains only points to the right of L:

      IF $x(p_k) > A$ THEN
             FOR all $(p_i,p_j)$ in SMALL $\cup$ EQUAL insert $p_i$ into CANDIDATES.
             FOR all $(p_i,p_j)$ in LARGE insert $p_i$ and $p_j$ into CANDIDATES.

9.  RETURN( BRIDGE(CANDIDATES,A) ).
```

Theorem 3.1

The function BRIDGE outlined above correctly determines the left and right bridge point in $O(n)$ worst case time and space.

Proof

The correctness of the algorithm can be seen from the following argument. As long as S contains more than three points, BRIDGE either finds the bridge in step 6 or discards redundant points of S applying the rules of Lemma 3.1 (steps 3,7,8) and calls itself recursively with a smaller pointset. Using the linear time median algorithm of Blum et al. [1, pp. 97-99], the body of BRIDGE without the recursive call can be executed in linear time and space. Furthermore, at least one quarter of the points of S are eliminated and not contained in CANDIDATES. Thus the worst case time and space requirements for the algorithm are bounded by

$$f(n) = \begin{cases} O(1) & n \leq 3 \\ f(3n/4) + O(n) & n > 3 \end{cases}$$

But it is well known that such a recursive function is O(n) [1, p.64].

Q.E.D.

Before closing this section we want to mention that our bridge finding algorithm was inspired by the linear time two variable linear programming algorithm of M. Dyer [3]. A closer look even shows that the bridge problem can be formulated as a linear programming problem. However, it is such a special case that it seemed worthwhile to spell out the simple algorithm explicitly instead of just referring to the more complicated general linear programming algorithm.

CONCLUSION

The main result of this paper is a new planar convex hull algorithm with worst case time bounds O(nlogH). Our method, contrary to previous algorithms, is not only worst case optimal but its performance is also sensitive to H, the number of vertices found on the hull.

As we have noted the construction of a non-trivial lower bound for the convex hull problem dependent on both n and H remains an open problem. At present an O(n+Hlogn) worst case time algorithm, for instance, is not totally inconceivable. Though we conjecture that the problem requires Ω(nlogH) time, it remains to be seen whether our method is indeed the "ultimate" planar convex hull algorithm.

REFERENCES

[1] Aho, A.V., Hopcroft, J.E., and Ullman, J.D.
The Design and Analysis of Computer Algorithms.
Addison-Wesley, Reading, MA, 1974.

[2] Avis, D.
On the Complexity of Finding the Convex Hull of a Set of Points.
McGill Univ., Montreal, School of Computer Science,
Report SOCS 79.2; 1979.

[3] Dyer, M.E.
Two-Variable Linear Programs are Solvable in Linear Time.
Manuscript; Dept. of Mathematics and Statistics, Teeside
Polytechnic, Middlesbrough, Cleveland, UK; 1982.

[4] Eddy, U.F.
A New Convex Hull Algorithm for Planar Sets.
ACM Transactions on Mathematical Software 3; pp. 398-403 and
pp. 411-412; 1977.

[5] Emde Boas, P.v.
On the Ω(nlogn) Lower-Bound for Convex Hull and Maximal Vector Determination.
Information Processing Letters 10, pp. 132-136; 1980.

[6] Graham, R.L.
An Efficient Algorithm for Determining the Convex Hull of a Finite Planar Set.
Information Processing Letters 1, pp. 132-133; 1972.

[7] Jarvis, R.A.
On the Identification of the Convex Hull of a Finite Set of Points in the Plane.
Information Processing Letters 2, pp. 18-21; 1973.

[8] Kung, H.T., Luccio, F., and Preparata, F.P.
On Finding the Maxima of a Set of Vectors.
Journal of the ACM 22, pp. 469-476; 1975.

[9] Preparata, F.P.
An Optimal Real Time Algorithm for Planar Convex Hulls.
Communications of the ACM 22, pp. 402-405; 1979.

[10] Preparata, F.P. and Hong, S.J.
Convex Hulls of Finite Sets of Points in Two and Three Dimensions.
Communications of the ACM 20, pp. 87-93; 1977.

[11] Shamos, M.I.
Computational Geometry.
Yale University, New Haven, Connecticut; Ph.D. Thesis; 1978.

[12] Yao, A.C.
A Lower Bound to Finding Convex Hulls.
Journal of the ACM 28, pp. 780-789; 1981.

ELIMINATING HIDDEN-LINES FROM MONOTONE SLABS.

DAVID RAPPAPORT
School of Computer Science
McGill University
Montreal, Quebec H3A 2K6

ABSTRACT

An algorithm is presented to solve a three dimensional hidden-line problem in linear time. The geometric properties of a restricted class of polyhedra is used to obtain and prove the result.

1. INTRODUCTION

A fundamental problem in displaying solid objects is to remove lines, that are obscurred by portions of the scene nearer the viewer. This is known as the hidden-line problem, or when shaded pictures are produced the hidden-surface problem.

In two dimensions optimal solutions exist for the visibility of a polygon from an edge or from a point [1], [2]. There are many solutions to the hidden-line and hidden-surface problems, motivated by a variety of applications, therefore there are a number of fundamentally different approaches to solve this problem. However no algorithm has been shown to be optimal, in fact no non-trivial lower bound is known. A common trait among algorithms for hidden-line elimination is that they exhibit a quadratic increase in time as the size of the input grows [7]. It is the intent here to introduce a restricted class of input and an algorithm that exhibits a linear worst case complexity. The use of a restricted class of input, has been employed to achieve assymtotic improvement in complexity as well as simplifying the problem sufficiently to permit theoretic analysis of the algorithm. The class of input is still general enough to conform to the requirements of a number of applications.

2. DEFINITIONS

A slab is the cartesian product of a simple polygon in two dimensions with an interval (a,b) along the z-axis. See figure (1). The simple polygon A and its twin B will be known as defining polygons of the slab. The vertices of the polygon A and B will be denoted $a_1 \ldots a_n$ and $b_1 \ldots b_n$ respectively and the edge (a_i,b_i) $1 \leq i \leq n$ is an edge of the slab. The x, y and z coordinates of a vertex a_i will be represented by $a_i(x)$, $a_i(y)$ and $a_i(z)$ respectively. This type of object is similar to the elementary object as described by Yao [8]. A monotone polygon is a simple polygon in which there exists a given direction defined by two vertices such that they are connected by two polygonal chains monotonic in that given direction. A monotone slab is then a slab with defining polygons that are monotone. Given a defining polygon with n edges, a slab will have n+2 faces, n of which are rectangles, or lateral faces. The remaining two faces are defining polygons. The faces are named F_1, F_2, ... F_{n+2}, where F_{n+1} and F_{n+2} are the defining polygons. The face F_i will represent the face with vertices $(a_i,a_{i+1},b_{i+1},b_i)$ $1 \leq i \leq n$. Note addition on vertex indices is performed modulo n. A parallel projection is used to project the figure onto the projection plane. Parallel projections are defined by their direction of projection. The direction of projection is normal to the projection plane. For simplicity of discussion a projection plane whose normal passes through the point (c,−c,c) for any constant c will be assumed throughout. The monotone slab will also have the origin of the coordinate system in its interior. The defining polygons will be assumed to be monotone with respect to the x axis. This configuration will be known throughout as the basic figure (BF).

3. CLASSIFICATION OF FACES AND EDGES

A face of a polyhedron can be classified as not visible (NV) if the interior of the polyhedron and the viewing position lie on the same side of the face. Otherwise the face is potentially visible (PV). The classes of the faces can be determined in linear time [4]. Class 1 edges are found at the intersection of two NV faces and these edges are not visible. Class 2 edges and class 3 edges are found at the intersection of an NV and a PV face. In the former case the interior dihedral angle is greater than 180 degrees and the edge is not visible in the latter the interior dihedral angle is less than 180 degrees and the edge is potentially visible. Class 4 edges are found at the intersection of two PV faces, these edges are potentially visible. Class 2, and 3 edges are known as boundary edges because they are found at the boundary of PV and NV faces. Refering to figure (1) the face $F_2 = (a_2,a_3,b_3,b_2)$ is an NV face and $F_1 = (a_1,a_2,b_2,b_1)$ is a PV face, (b_2,b_3) is a class 1 edge, (a_4,b_4) is a class 2 edge, (a_2,b_2) is a class 3 edge and (a_1,b_1) is a class 4 edge.

4. THE COVER RELATION

Given a viewing position Q, face F_j covers face F_i if and only if there exists a point p_i in F_i such that (p_i,Q) intersects the interior of the face F_j at a point p_j. Note: Q is a point at infinity lying on any line normal to the projection plane. Furthermore (p_i,p_j) is not in the interior of the slab. From the definition F_i is a PV face and it can be shown that F_j is an NV face [6]. If all cover relations in a slab are determined then the hidden lines of the slab can be found as well.

There are two extreme vertices of the monotonic chains on the defining polygon A. Choose the vertex that lies nearer the viewing position as a starting vertex. The opposite extreme vertex is the terminal vertex. To determine these extreme vertices in a monotone polygon a linear algorithm due to Preparata and Supowit may be used [5]. Two sequences of lateral faces can be created. Begining at the starting vertex and taking successive lateral faces until the terminal vertex is reached will comprise the faces of one sequence. If the lateral faces are found on the upper part of the BF (the up direction is arbitrary), the sequence is known as the upper chain (UC). The lateral faces of the lower part of the BF ordered from the starting vertex to the terminal vertex form the faces of the lower chain (LC).

5. PRELIMINARY RESULTS

Lemma 1: If F_j covers F_i then F_i and F_j both appear in the LC (UC) sequences and F_j precedes F_i in the sequence.

Proof: Let p_{j*} be on the surface of F_j and let p_{i*} be on the surface of F_i. First assume F_i and F_j are both in LC but F_j follows F_i in the sequence. By the monotone property $p_{j*}(x) < p_{i*}(x)$ which violates the cover condition i.e. the existence of a line through the points p_{i*},p_{j*},Q where $Q(x) > p_{j*}(x) > p_{i*}(x)$.

Now let F_j be in LC and let F_i be in UC. Construct a line by joining the points Q, p_{j*}, p_{i*} where p_{j*} is on the surface of F_j and P_{i*} is on the surface of F_i. See figure (2a). By definition of the cover relation the line segment (p_{j*}, p_{i*}) is not contained in the interior of the BF. Construct a line segment L1 by joining the points a_{j*}, p_{j*}, b_{j*} normal to the line segment (p_{j*}, p_{i*}) and a line segment a_{i*}, p_{i*}, b_{i*} parallel to L1 where a_{j*}, a_{i*}, b_{i*}, b_{j*}, are on edges (a_j, a_{j+1}), (a_i, a_{i+1}), (b_i, b_{i+1}), (b_j, b_{j+1}). By joining a_{j*} to a_{i*} and b_{j*} to b_{i*} we construct a parallelogram with the vertices a_{j*}, b_{j*}, a_{i*}, b_{i*}. See figure (2b).

Because F_i is on UC and F_j on LC there exists a path of vertices a_i, a_{i+1}, ... a_f, ... a_{j-1}, a_j, ... a_l, ... a_{i-1}. a_f represents the point on face A with maximium x coordinate and a_l represents the point on face A with the minimum x coordinate. Since $a_f(x)$ is greater than $a_j(x)$ then we can show that the line segment we constructed (a_{i*}, a_{j*}) intersects this path. Let the point of intersection be a_{k*}. Similarly a path b_i, b_{i+1}, ... b_f,... b_{j-1}, b_j, ... b_l intersects (b_{i*}, b_{j*}) at b_{k*}. Joining a_{k*} to b_{k*} we find this line intersects the line segment (p_{i*}, p_{j*}) in our constructed parallelogram at the point p_{k*}. Clearly a point arbitrarily close to p_{k*} can be found interior to the BF and on the line segment (p_{i*}, p_{j*}) therefore F_i cannot cover F_j. Q.E.D.

The following lemma leads to a method where the classification of faces and edges are used to determine cover relations in the BF.

Lemma 2: Let F_i be a covered face and suppose F_i appears before any other covered face in the LC sequence. Then F_i is covered by F_{i-1} and faces F_i, F_{i-1} have the edge (a_i, b_i) a class two boundary edge in common.

Proof: It is clear that when examining faces in LC the first candidate to be covered is the first PV face in the sequence that appears after at least one NV face. Let the PV face be F_i and the NV face F_{i-1}. We can choose points p_{i-1} and p_i on faces F_{i-1} and F_i respectively. Constructing the line (p_i, p_{i-1}) we see that it is not in the interior of the BF since the line lies between two adjacent faces on the same monotone chain. By picking a point u on (a_i, b_i) we can construct the triangle $T=(p_i, p_{i-1}, u)$. The angle formed between (p_i, u) and (p_{i-1}, u) is clearly less than 180 degrees therefore the interior angle is greater than 180 degrees and (a_i, b_i) is a class two boundary edge. See figure (3). If we extend the line (Q, p_{i-1}) we get an intersection point p_{i*}, with the half plane that F_i lies on. Choosing p_{i-1} close enough to (a_i, b_i) we see that we can have (Q, p_{i*}) intersect the face F_i. Therefore F_{i-1} covers F_i. Q.E.D.

For display purposes we are concerned about the planar representation of the basic figure. It can be shown that the parallel projection is an affine transformation and can be represented by the following equations: [3]

$$x' = r_1x + s_1y + t_1z; \quad y' = r_2x + s_2y + r_2z; \quad z' = r_3x + s_3y + r_3z$$

To decide which vertex of the BF is 'nearer' Q it is sufficient to compare the z' coordinates. If p_i lies on a line segment (Q,p_j) p_i,p_j in the BF then p'_i, p'_j are coincident. Consequently if F_j covers F_i the projected representations of the two faces F_j and F_i necessarily intersect. Another important feature when using the parallel projection is that the monotonicity of the defining polygons is retained. The projection of A, A' is monotone with respect to the x' axis.

Let F'_i be the parallel projection of F_i. Similarly the point p'_i is the projection of p_i. The BF has the following properties. $a'_i(x')$ is always greater than its twin $b'_i(x')$ and $a'_i(y')$ is always less than $b'_i(y')$.

Let us now examine a monotone slab that contains a single class 2 edge (a_i,b_i) on a face in the LC sequence. Consider two subsets of the LC sequence:

$$O = (F_{i-1}, F_{i-2} \dots F_{i-m});\ U = (F_i, F_{i+1} \dots F_{i+n})$$

Every face contained in O is an NV face, and F_{i-m-1} is a PV face. Every face in U is a PV face with F_{i+n+1} an NV face. As a direct consequence of Lemma 1 all covered faces in the BF are in the set U and all covering faces in are contained in O. For this reason O is known as the 'over' set of faces and U the 'under' set of faces.

Two chains of projected boundary edges can now be constructed.

$$\begin{aligned} C(O) = (\ & (a'_i,a'_{i-1}), (a'_{i-1},a'_{i-2}) \dots \\ & (a'_{i-m+1},a'_{i-m}), (a'_{i-m},b'_{i-m})\ ) \\ C(U) = (\ & (b'_i,b'_{i+1}), (b'_{i+1},b'_{i+2}) \dots \\ & (b'_{i+n-1},b'_{i+n}), (b'_{i+n},b'_{i+n})\ ) \end{aligned}$$

Refering to figure (1), i=4, m=2, n=3. The chain C(O) and C(U) are a sequence of all the boundary edges from faces in O and U, ordered in C(U) away from the viewing position and monotone decreasing, and in C(O) toward the viewing position, and monotone increasing.

Lemma 3: C(U) and C(O) intersect exactly once.

Proof: In Lemma 2 it was shown that F_{i-1} covers F_i. As a direct consequence the intersection of the interior of F'_{i-1} and F'_i is non-empty, and therefore the intersection of the interior of O and U is non-empty. This necessitates the intersection of C(O) and C(U). Only a single intersection occurs due to the monotonicity of C(O) and C(U). Q.E.D.

A procedure can now be introduced to determine the intersection of the chains C(O) and C(U) given a class two boundary edge in the BF.

```
procedure COVER(j,k)
comment:
$j \leftarrow i$, $k \leftarrow i+1$ when $(a_i,b_i)$ is a class 2 edge.
```

```
begin
 repeat
1.if $(b'_k, b'_{k+1})$ intersects $(a'_j, a'_{j-1})$ then exit
2.    else if $(a'_j, a'_{j-1})$ intersects $(a'_{k+1}, b'_{k+1})$
3.      and $(a'_{k+1}, b'_{k+1})$ is of class 3
4.        then exit
5.          else k ← k+1;
6.      else if $(b'_k, b'_{k+1})$ intersects $(a'_{j-1}, b'_{j-1})$
7.          and $(a'_{j-1}, b'_{j-1})$ is of class 3
8.            then exit
9.              else j ← j-1;
 until forever

end.
```

Lemma 4: Procedure COVER determines the edges where C(U) and C(O) intersect.

Proof: If the algorithm continues if and only if the point a'_j lies interior to a projected face in U and b'_k lies interior to a projected face in O, then the intersection of C(O) and C(U) will be detected. It will be shown inductively that these conditions hold prior to the execution of every iteration of COVER.

When COVER is entered the induction hypothesis is true as a direct consequence of lemma 2. Assume at the rth iteration the hypothesis is true. If the condition of line 1. is satisfied or the condition in line 2. and in line 3. are both satisfied or the condition of line 6. and of line 7. are both satisfied The intersection of C(O) and C(U) is found and we are done. Examine the other two cases. When k is assigned a value of k+1 the entire edge (b'_k, b'_{k+1}) must lie in the interior of face F'_{i-1}. Similarly when i is decremented the edge (a'_j, a'_{j-1}) lies within the interior of the face F'_k. Q.E.D.

Let p' be the point of intersection of C(U) and C(O).

Lemma 5: Given a BF with a single class 2 edge in LC. A point q' on C(U) is not visible if and only if q' precedes p' on C(U).

Proof: If q' precedes p' in C(U) then q' is found in the interior of the intersection of U and O, otherwise q' is not contained in this intersection. Since faces in O are the only covering faces in the BF all other points in C(U) are visible. Q.E.D

Corollary: The edges (a'_i, b'_i)contained in C(U) for all i where b'_i precedes p in C(U) are intersected by C(O) and are hidden at points interior to faces in O.

Procedure COVER can be used to determine the hidden lines in a BF as long as only one class 2 edge is found per LC (UC) sequence. The general case where a number of class 2 edges are found per LC (UC) sequence will now be examined.

Consider a BF with d class 2 boundary edges in LC (UC). The faces of the sequence can be grouped in the sets:

$$O_1 = (F_{i_1-1}, F_{i_1-2} \dots F_{i_1-m_1});\ U_1 = (F_{i_1}, F_{i_1+1} \dots F_{i_1+n_1})$$

$$O_d = (F_{i_d-1}, F_{i_d-2} \dots F_{i_d-m_d});\ U_d = (F_{i_d}, F_{i_d+1} \dots F_{i_d+n_d})$$

Refering to figure (4), $d=2, i_1=3, m_1=1, n_1=3, i_2=7, m_2=1, n_2=3$. The properties of the faces of O_i and U_i 1 i d is as described above for O and U. Similarly the chains $C(O_1)$, $C(O_2)$, ... $C(O_d)$, and $C(U_1)$, $C(U_2)$, ... $C(U_d)$ can be constructed.

In this situation a face in U_i may be covered by a face in O_j if and only if $i \geq j$.

This follows directly from lemma 1.

Lemma 6: A face in O_{i-1} covers a face in U_i if and only if $C(O_{i-1})$ intersects $C(O_i)$.

Corollary: The faces in O_{i-1} that cover any face in U_i must lie on an edge that is equal to or follows the intersected edge in $C(O_{i-1})$.

The proof is omitted but may be found in [6].

In general if a face F'_h in U_i is covered by a face F'_g in O_j where j is less than i, then $C(O_j)$ intersects $C(O_i)$ or $C(O_j)$ intersects $C(O_k)$ $i > k > j$ and $C(O_k)$ intersects $C(O_i)$.

From the previous discussion it should be clear that an efficient method is required to find the class 2 edges and to create the chains $C(U_1)$, $C(O_1)$ etc. The following lemma proves an important result concerning boundary edges in slabs. This result applies to slabs in general, not just monotone slabs. The results of this lemma leads to a method of traversing through the boundary edges.

Lemma 7: Boundary edges in slabs form one simple cycle.

Proof:

Every edge e_i in a slab has exactly four neighbouring edges. By examining an arbitrary boundary edge, e_b one can easily show that at each endpoint of e_b there is exactly one neighbouring boundary edge.

Let edges e_1 and e_2 be neighbours of e_b at one of its endpoints and let edges e_2 and e_3 be the neighbours of e_b at the other endpoint.

Six faces can be described by taking adjacent pairs of edges, of these six faces four are unique. These four faces numbered F_1 to F_4 are listed below.

$F_1 - e_1, e_2$; $F_2 - e_3, e_4$; $F_3 - e_1, e_b$ or e_3, e_b;
$F_4 - e_2, e_b$ or e_4, e_b

Only one of F_3, F_4 can be a PV face because we defined e_b a boundary edge. F_1 and F_2 can both be PV faces, both be NV faces or one can be NV the other PV. Choosing F_1, F_2 and F_4 PV faces and F_3 an NV face e_b has boundary edge neighbours e_1 and e_3. If F_1 and F_3 are chosen to be NV faces and F_2 and F_4 PV faces then e_b has boundary edge neighbours e_3 and e_2. The other combinations of face class all yield the result of two boundary edge neighbours for e_b.

It remains to show that no disjoint cycles exist in a slab.

Consider the points a_i and b_i. At least one of them must lie on a boundary edge, because if face F_i is PV then the edge (b_i, b_{i+1}) is a boundary edge. If F_i is NV then (a_i, a_{i+1}) is a boundary edge. If both a_i and b_i lie on boundary edges then the lateral edge (a_i, b_i) is a boundary edge. Using the above two observations we see that when traversing the vertices by boundary edges the sequence of vertices visited is of the following form:

$\ldots a_{i-1}, a_i, a_{i+1} \ldots$
or $\ldots a_{i-1}, a_i, b_i, b_{i+1} \ldots$
or $\ldots b_{i-1}, b_i, a_i, a_{i+1} \ldots$
or $\ldots b_{i-1}, b_i, b_{i+1} \ldots$

Traversing the vertices this way visits all the boundary edges once. Q.E.D.

6. THE ALGORITHM

In this section an algorithm is presented that computes the lines to be displayed of a parallel projection of the basic figure.

Recall that a monotone polygon has two extremal vertices, which were refered to in section 4 as the starting and terminal vertices.

Algorithm ELIMINATE:

Input: The projected vertices, edges,and faces of the BF and the edge and face classes. An arbitrary projection plane may be assumed. Output: A list of the visible edges.

Method:

Begining at the starting vertex for the LC and UC sequences do

step 1: While the terminal vertex is not reached, traverse the boundary edges until a class 2 edge is found.Denote the class 2 edge (a_i, b_i). $j \leftarrow i$, $k \leftarrow i+1$.

Step 2: Call procedure COVER(j,k) and update the list of edges as specified in lemma 5. (Note: COVER exits with modified values of j and k.)

Step 3: Let the intersected edge in C(O) be denoted by EO and let the point of intersection be denoted by p.

```
 if the updated edge EO is not visible at the point p then

     push j onto top of STACK
     go to Step 1;
   else
       pop j from the top of  STACK
       go to Step 2;
```

7. ANALYSIS OF THE ALGORITHM

The main result of this paper is stated.

Theorem: Algorithm ELIMINATE computes the hidden-lines of a monotone slab in time proportional to the number of its edges.

Proof: The results of section 5 lead to the correctness of the algorithm. Details may be found in [6]. The following arguments show that ELIMINATE runs in linear time.

Boundary edges can be traversed in linear time using the result of lemma 7. Let h_i be the number of edges in $C(O_i)$ and k_i the number of edges in $C(U_i)$. Let d be the number of class 2 edges in LC and let m be the number of boundary edges in the LC sequence. Let n be the number of edges in the BF. When processing LC procedure COVER performs a maximum of

$$3\left(\sum_{i=1}^{d} (h_i + k_i)\right) \qquad (1)$$

edge intersections if the STACK is never popped, since it is called, once per class 2 edge. (1) is equal to O(m). Every time the stack is popped only one edge is re-examined. The STACK is popped a maximum of d times so this adds only a constant amount of work. Note: COVER can be easily modified so that edge visibility updates are performed when an edge is visited so this step does not add to the complexity of the algorithm. Assume the UC sequence has r boundary edges. Since LC and UC are disjoint and processed independently

and $\quad r + m < n$

therefore $\quad O(r + m) < O(n)$

So ELIMINATE has a complexity of O(n). Q. E. D.

8. CONCLUSION

Yao [8] introduced a class of input, that was equivalent to a set of disjoint slabs in a scene, known as an elementary object. An O(n logn) hidden-surface algorithm was presented for elementary objects in [8]. In this paper an algorithm was introduced to compute the visible lines of a monotone slab in linear time. Since the output is linear, the lower bound for this problem is linear and therefore the algorithm is optimal. A linear, and optimal hidden-surface algorithm is presented in [6] using the monotone slab class of input.

ACKNOWLEDGEMENTS

The author would like to thank S. Zucker and the Computer Vision and Graphics Lab of the department of Electrical Engineering at McGill University for providing the facilities to implement the algorithm presented here. Special thanks to David Avis for his help and encouragement throughout the course of research on this problem.

REFERENCES

1. Avis, D. and G.T. Toussaint, An Optimal Algorithm for Determining the Visibility of a Polygon from an Edge. IEEE Trans. on Comput. Vol. C-30, No. 12, (Dec. 1981)

2. El Gindy, H. and D. Avis, A Linear Algorithm for Computing the Visibility Polygon from a Point. Journal of Algorithms 2, 186-197 (1981)

3. Klein, F., Elementary Mathematics from an Advanced Standpoint, Geometry, Dover, New York 1939

4. Loutrel, P.P., A Solution to the Hidden-Line Problem for Computer-Drawn Polyhedra. IEEE Trans. on Computing C-19 3, (March 1970), 205

5. Preparata, F.P. and K.J. Supowit, Testing a Simple Polygon for Monotonicity. Information Processing Letters, 12, 4,(August 1981)

6. Rappaport, D.H., Visibility in Restricted Classes of Polyhedra. M.Sc. Thesis, McGill University (in preparation)

7. Sutherland, I. E., R.F. Sproull and R.A. Schumacker, A Characterization of Ten-Hidden Surface-Algorithms, Computing Surveys Vol. 6 No. 1 March 1974.

8. Yao, F. Frances On the Priority Approach to Hidden-Surface Algorithms. Proc. 21st Annual Symposium on Foundations of Computer Science, Oct. 1980.

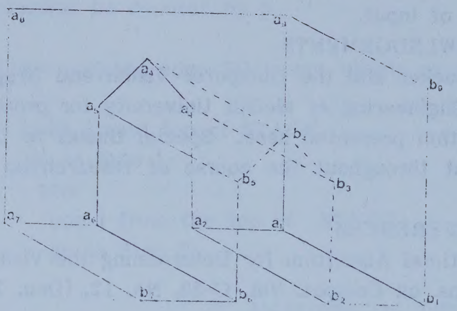

a₀
a₁
b₀
a₃
a₄
a₅
a₆
a₇
a₂
a₁
b₅
b₃
b₄
b₇
b₆
b₂
b₁

Figure 1.

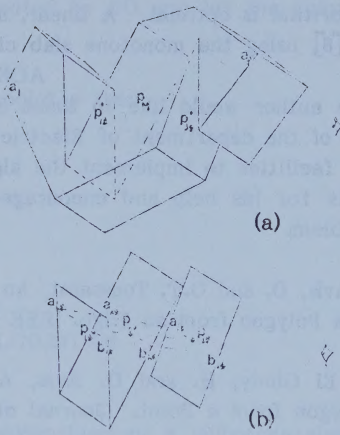

a₁
a₂
P₁₂
P₂₁
P₂'
V
(a)
a₁₄
a₂
P₂
P₁₄
b₁₄
a₁
P₁₄
b₂
b₁₄
V
(b)

Figure 2.

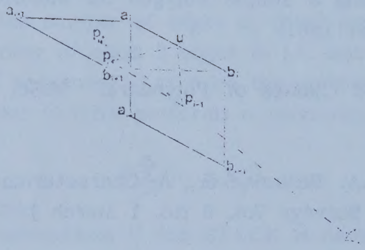

a_{i-1}
a_i
p_i
u
p_i
b_{i-1}
b_i
p_{i-1}
a_{i+1}
b_{i+1}
V

Figure 3.

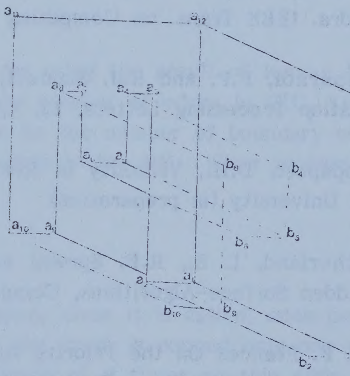

a₁₁
a₁₂
a₈
a₇
a₅
a₄
b₁₂
a₆
a₃
b₈
b₄
a₁₀
a₉
b₅
b₆
a₂
a₁
b₁₀
b₉
b₁
b₂

Figure 4.

Minimum Edge Length Partitioning of Rectilinear Polygons[†]

(Extended Abstract)

Andrzej Lingas*
Ron Y. Pinter*
Ronald L. Rivest*
Adi Shamir**

Abstract

We consider the problem of partitioning a rectilinear polygon into disjoint rectangles using the minimum amount of "ink". The given polygonal figure may or may not have holes in it, and the complexity of the problem is shown to depend on this very fact: For hole-free figures, we provide a polynomial-time algorithm, whereas for figures containing holes the problem is shown to be NP-complete.

For the hole-free case, the time complexity of the partitioning algorithm is $O(n^4)$ for arbitrary rectilinear shapes, but for non-trivial special cases the complexity can be reduced to $O(n^3)$. In the general case it turns out that even degenerate holes, *i.e.* points through which the decomposing edges have to go, are enough to render the problem intractable.

More interestingly, the proof techniques for the NP-completeness results can be applied to an even more general case, namely where the polygons are no longer rectilinear and a partitioning into convex polygons with minimum edge length is sought.

1. Introduction

The problem of dividing a polygonal region into parts has received considerable attention recently ([ChDo79], [LLMP79], [LLMPL79], [GJPT78]). Traditionally, convexivity was the overriding issue and the goal was to obtain a convex partition using the minimum number of components. In this paper we focus on *three* new aspects of the problem, two regarding shape, and one — the minimality criterion: First we consider only *rectilinear* polygons, *i.e.* figures with "vertical" and "horizontal" boundary edges; such polygons can always be *partitioned* into rectangles (with the same orientation as the original figure). Second we allow the inclusion of *holes* (or *islands*) in the given figure, *i.e.* one enclosing polygon may contain several polygonal shapes cut out of it (see Figure 1). Finally, we introduce and investigate a new minimality criterion, namely — the total length of the *edges* (lines) used to form the internal partition; we call this criterion *minimization of edge length.*

[†] This research was supported by DARPA grant N00014-80-C0622, NSF grant MCS-8006938, and an IBM graduate fellowship.

* Laboratory for Computer Science, Massachusetts Institute of Technology, Cambridge, Mass. 02139

** Department of Applied Mathematics, The Weizmann Institute of Science, Rehovot 76100, Israel.

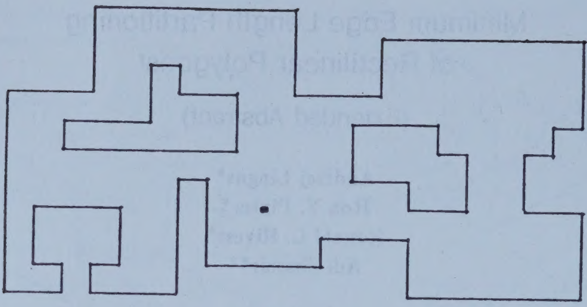
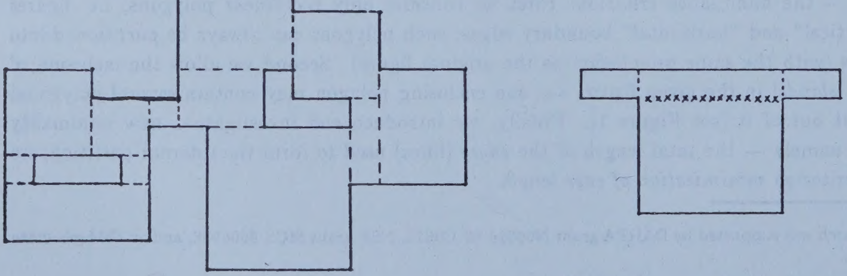


Figure 1. A rectilinear figure with three holes (one is degenerate).

We investigated various versions of the problem of minimizing the edge length partitioning of a rectilinear figure. The major result of this paper is that the critical characteristic of an instance of the problem is whether it has holes or not: If it does not, a polynomial time algorithm exists to solve it optimally; otherwise, the problem is NP-complete ([GaJh79]) even if the holes are degenerate, *i.e.* they are points through which the partitioning edges have to pass. Interesting polynomial-time heuristics that can achieve solutions whose edge length is within a constant factor from optimal are discussed in [Lin81].

The problem of finding optimal partitions of such figures with respect to our criterion has applications in several domains: Process control (stock cutting), automatic layout systems for integrated circuits (channel definition), and even architecture (internal partitioning into offices). The *minimum edge-length* partition is a natural goal for these problems since there is a certain amount of waste (e.g. sawdust) or expense incurred (e.g. for dividing walls in the office) which is proportional to the sum of edge lengths drawn. For VLSI design, this criterion is used in the MIT "PI" (Placement and Interconnect) System ([Ri82]) to divide the routing region up into channels – we find that this produces large "natural-looking" channels with a minimum of channel-to-channel interaction to consider. At an intuitive level, these partitions might be considered analogous to the way rectilinear soap-bubbles would decompose a given space into convex parts.

Notice that the traditional minimality criterion of reducing the *number* of components — which leads to different applications — does not coincide with ours. Figure 2(b) shows a trivial (and generic) example of this phenomenon. Also, the relative difficulties of the various classes of problems turn out to be different, as discussed in [Lin82].



(a) Holes interact with boundary

(b) Minimum number of rectangles (xxxxx) does not coincide with minimum edge length (---).

Figure 2. Minimum edge length decompositions into rectangles.

In what follows, we first set up some notation (Section 2). Then we present the polynomial-time algorithm for the hole-free cases (Section 3) and sketches of the NP-completeness proofs for the general case (Section 4). Finally, we pose extensions and open problems (Section 5).

2. Notation and Definitions

A rectilinear *boundary* is a simple polygon, all of whose sides are either parallel or perpendicular to each other. More precisely, each side is a segment of a line in the plane, and the set of lines corresponding to a boundary can be partitioned into two sets such that within each set all lines are parallel to each other, and the two sets are mutually perpendicular. We arbitrarily call one direction *horizontal*, the other *vertical*. Clearly, all *corners* of a boundary are either of 90^0 (*convex*) or 270^0 (*concave*). A boundary may be represented by a list of its corners as points in the plane, $P_1, P_2, \ldots, P_n$, in (say) clockwise order. (Simplicity can then be checked in $O(n \log n)$ time ([ShHo76])).

A rectilinear *hole* in a rectilinear boundary is a simple rectilinear polygon fully enclosed by it. We allow only holes whose orientation is the *same* as that of the enclosing boundary, *i.e.* the horizontal and vertical directions corresponding to the polygons coincide. A hole is represented as before by a list of points describing its circumference; a hole is called *degenerate* if it consists of a single point.

A rectilinear *figure* is a set of rectilinear boundaries, each of which may contain an arbitrary number of non-overlapping holes in it. Notice that holes are not allowed to contain further holes. From here on, we shall consider (without loss of generality) only figures consisting of one boundary and its holes; for the problems considered in this paper, an optimal solution for a set of boundaries may be obtained by solving the problem for each member separately. Figure 1 shows a typical figure consisting of a boundary with several holes in it.

A *rectangular partition* of a figure is a set of line segments, $l_1, l_2, \ldots, l_m$, lying within its boundary and not crossing any holes, so that when drawn into the figure the area not enclosed by holes is partitioned into rectangles; clearly, no "loose ends" are allowed. Figure 2(a) is an example of such a partition. We could have given a formal definition of what it means for a set of line segments to partition an area (with holes) into rectangles, but it would not serve any useful purpose in the extext of this abstract.

Finally, the partitioning line segments are called *edges*. A minimum *edge length* partitioning of a figure is a rectangular partitioning such that the sum of lengths of its edges is smallest among all valid partitionings. Incidentally, this is equivalent to requiring that the total of all perimeters of the resulting rectangles be minimized.

3. A Polynomial-time Partitioning Algorithm for Hole-free Figures

In this section we show that a minimum edge length partitioning of a rectilinear figure consisting only of a simple boundary (henceforth *hole-free*) can be found in polynomial time. The complexity in the general hole-free case is $O(n^4)$ (where n is the number of sides on the boundary), but in case the boundary has a shape of a *histogram* (to be defined in due course), an $O(n^3)$ algorithm exists.

The algorithm used makes use of dynamic programming (see, for example, [AHU74]) — a rather natural approach for this problem: minimum edge length triangulation can be easily solved using the this technique. The difficult part of the proof for the rectangular case involves showing how a given polygon can be split in a small number of ways while guaranteeing that the optimal partitioning is consistent with one of these splits.

Before we proceed, let us add two definitions used only in this section: First, the *grid* induced by a boundary is the set of horizontal and vertical lines (and their intersection points) on which segments of the boundary lie. Second, a *constructed line* is a maximal extension of a partitioning edge to include any sides of the boundary that are contiguous to the edge and go in the same direction. Two or more aligned edges may be put on the same constructed line, but this is not always the case. This is exemplified in Figure 3 where constructed lines are dotted, whereas edges are dashed.

The following two lemmas serve to center our attention on the crux of the matter:

Lemma 1. In all minimum edge length solutions for any given boundary $\mathcal{B}$, all edges and internal corners lie on the induced grid.

Lemma 2. For any given boundary $\mathcal{B}$, there exists a minimum edge length partitioning such that every constructed line has at least one end on the original boundary $\mathcal{B}$.

Ends of constructed lines that lie on the boundary are called *anchors*; thus Lemma 2 can be restated as saying that every constructed line has at least one anchor. An anchor may be a corner or a point on a side perpendicular to the edge. In Figure 3 we marked anchors with solid circles; in 3(a) an optimal solution satisfying the lemma is shown, whereas in 3(b) we see that there exist figures for which all constructed lines can have at most one anchor in order to obtain optimality. Thus Lemma 2 cannot be strengthened.

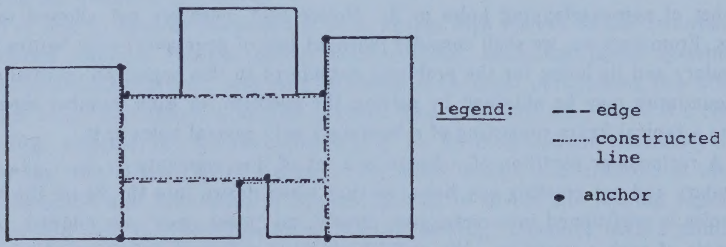


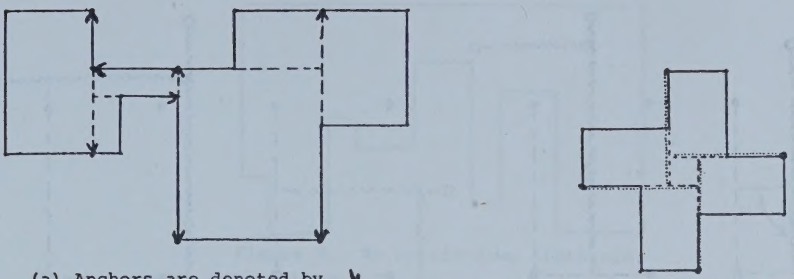
Figure 3. The constructed lines of a decomposition.

In the process of finding the optimal solution, we shall look at portions of the original figure which are in themselves hole-free boundaries. We claim that it is enough to look at portions whose boundary consists of a contiguous piece of the original boundary and *at most* two constructed lines, that again, are contiguous. In other words, the most complicated intermediate figure looks like a piece of the original boundary cut off by a (convex or concave) L-shaped pair of constructed edges; sometimes it is just cut off by one edge, and initially it is not cut off at all. Figure 5 shows two such intermediate figures. We shall show that our algorithm generates only intermediate figures satisfying the above property, and this in fact is the invariant preserved by all partitioning rules.

Now we have to devise the rule for partitioning: For each intermediate figure we define a *candidate* point depending on the number of constructed lines it has: If 0, then any convex corner may be chosen; if 1, then either end of the constructed line is chosen; and if 2, then we have to choose the common end of the two constructed lines (remember they are adjacent).

The *matching* point is defined as one lying on the induced grid, and such that there exists a (not necessarily optimal!) partitioning of the partial figure in which the candidate point and the matching point are kitty corners of a rectangle. Figure 5 shows forbidden cases of matching points. Furthermore, in order to maintain optimality, the matching point is restricted to lie in certain areas with respect to the candidate point as characterized by Lemmas 3 and 4 below.

Note that without loss of generality, we can always assume that the partial figure has two



(a) Anchors are denoted by ↓

(b) A "pinwheel": only one anchor per const. line.

Figure 4. There exist optimal solutions in which every constructed line is anchored.

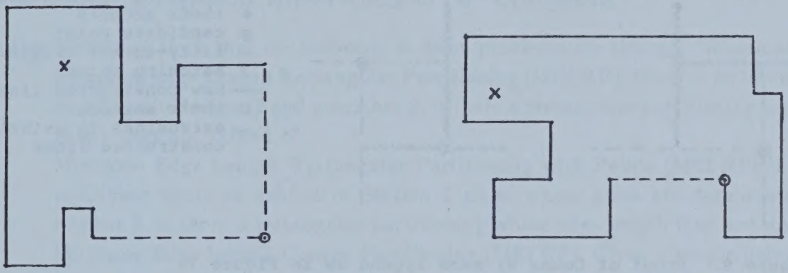


Figure 5. Forbidden matching points. ⊙ = candidate point
x = matching point

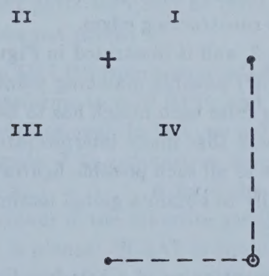


Figure 6. The four quadrants around the kitty-corner origin (+).

constructed lines and that the candidate point is at their common end. Now the candidate point can be either convex or concave with respect to the rest of the partial figure.

First let us assume it is convex. Then let us define the *kitty-corner origin* as the kitty corner of the rectangle that would have been formed by using the two constructed lines as its sides (see Figure 6). There are four quadrants defined with respect to this origin. We claim that

Lemma 3. The matching point of a convex candidate cannot reside in the quadrant including the candidate point (IV in the Figure 6). Whenever it resides in one of the other quadrants, the basic invariant is preserved.

The first part is proven by realizing that one edge will be superfluous in any solution placing the matching point in this quadrant (Figure 7(a)), thus defying its optimality. The second part is proven by inspection, showing that the only possible partitions that will not violate minimality

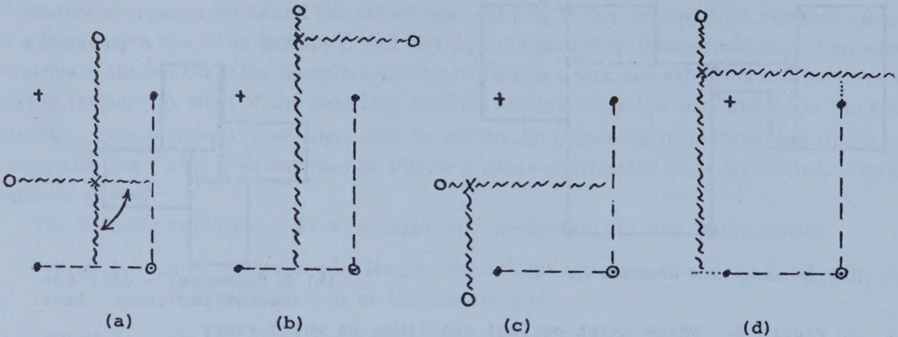
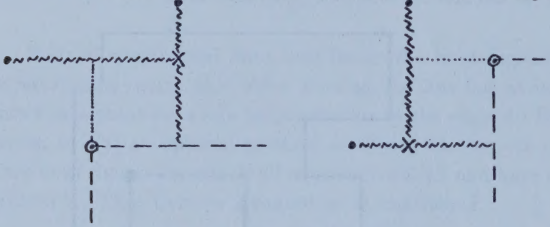


Figure 7. The proof of Lemma 3.

legend:
--- existing constructed line
• their anchors
⊙ candidate point
+ kitty-corner origin
x matching point
~~~ new constructed lines
o their anchors
—· extensions to existing constructed lines



Figure 8. Proof of Lemma 4; same legend as in Figure 7.

are as shown schematically in Figure 7(b),(c) and (d). Notice that constructed lines are extended in the process in a manner consistent with their definition.

If the candidate point is concave, the situation is simpler:

Lemma 4. The matching point of a concave candidate must have a horizontal or vertical projection on one of the constructing edges.

The proof is similar to that of Lemma 3, and is illustrated in Figure 8.

For each candidate point there are $O(n^2)$ possible matching points, and the (intermediate) minimum edge length partitioning resulting from each match has to be evaluated. But since the induced grid has $O(n^2)$ points, there are only that many intermediate figures looked at during the whole algorithm. If we maintain a table of all such possible figures (sorted, say, by area), we can apply dynamic programming successfully to obtain a global minimum.

Putting everything together, we obtain

Theorem 1. Minimum edge length partitioning of a hole-free figure can be found in time $O(n^4)$.

If the hole-less figure can be described by a list $P_1, P_2, \ldots, P_n$ such that the x-coordinates of the points form a monotonically non-decreasing sequence, or — alternatively — the y-coordinates have this property, then we say that the figure is a *histogram*. In this case, a faster algorithm exists, which is easiest to visualize when histograms are being drawn upside down, as in Figure 9; the long edge at the top is the *sky* and all sides parallel to it are *roof tops*.

We take the roof top closest to the sky, and at each of its corners (unless it is at the left or right end) we have to decide whether to draw a vertical or a horizontal edge. We can show that one of these edges (in its entirety) appears in a minimum edge length solution. Again, we use dynamic programming to keep track of optimality of partial figures. For convergence, notice that the sky is lowered for a contiguous portion of the histogram each time a horizontal edge is drawn, or we are left with a shorter histogram if a vertical edge is drawn. This procedure is faster than the general one, and we have
~~~

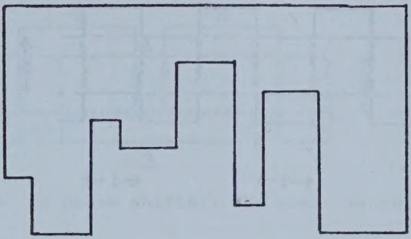


Figure 9. An upside-down histogram.

Theorem 2. Minimum edge length partitioning of a histogram can be found in time $O(n^3)$.

4. Partitioning Figures with Holes is NP-complete

In this section we show that the following decision problems are strongly NP-complete:

(1) **Minimum Edge Length Rectangular Partitioning (MELRP):** Given a rectilinear figure as defined in Section 2 and a number k, is there a rectangular partitioning whose edge length does not exceed k?

(2) **Minimum Edge Length Rectangular Partitioning with Points (MELRPP):** Given a rectilinear figure as defined in Section 2 all of whose holes are degenerate, and a number k, is there a rectangular partitioning whose edge length does not exceed k?

(3) **Minimum Edge Length Convex Partitioning (MELCP):** Given a simple polygon with non-overlapping, polygonal holes in it, and a number k, is there a partitioning of its area into convex parts whose edge length does not exceed k?

(4) **Minimum Edge Length Convex Partitioning with Points (MELCPP):** Given a simple polygon, a set of points within its area, and a number k, is there a partitioning of its area into convex parts such that all given points reside on decomposing edges whose edge length does not exceed k?

We chose to transform MELRP from planar satisfiability (PLSAT), shown to be NP-complete in [Lich82]. MELRP transforms to MELRPP, and the proofs for MELCP and MELCPP are analogous (with slight modifications). In fact, we use a slightly less restricted version of PLSAT defined as follows: A formula $\mathcal{F}$ in conjunctive normal form with exactly 3 literals per clause (3CNF), whose variables $x_i, i = 1, \ldots, n$ form the set X and clauses $c_j, j = 1, \ldots, m$ form the set C, is said to be *planar* if the bipartite graph $G(\mathcal{F}) = (X \cup C, E)$ with $E = \{ (x_i, c_j) \mid x_i \text{ or } \bar{x}_i \text{ is a literal in } c_j \}$ is planar. PLSAT is the problem of deciding whether a given planar formula $\mathcal{F}$ is satisfiable or not.

In order to transform PLSAT to MELRP we construct a rectilinear figure $\mathcal{H}$ such that there exists a partitioning of $\mathcal{H}$ into rectangles using edge length of at most k if and only if the given planar formula $\mathcal{F}$ is satisfiable. The key idea is to lay out $\mathcal{H}$ as an image of a planar circuit such that partitionings of $\mathcal{H}$ simulate computations of the truth value of $\mathcal{F}$.

The basic component of $\mathcal{H}$ is a crooked (geniculated) wire used to transmit 0 (false) and 1 (true) signals. Figure 10 shows a piece of a wire: Its sides are square waves whose phases are the same. A wire is most efficiently partitioned into rectangles (with respect to edge length) by a series of edges either all perpendicular or all parallel to its direction (see dashed lines in the figure). We interpret a parallel partitioning as a 1, and a perpendicular one as a 0. A wire changes the value of the signal it carries if it makes a 90^0 turn, but it may split (fan-out) without changing its value; see Figure 11 for the details.

$\mathcal{H}$ is constructed as a homeomorphic image of the graph $G(\mathcal{F})$. Each vertex in X corresponds

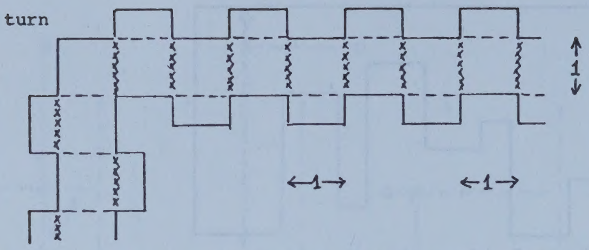


Figure 10. A piece of a crooked wire including a turn.
Two optimal decompositions are shown: – – –horizontal
xxxx vertical

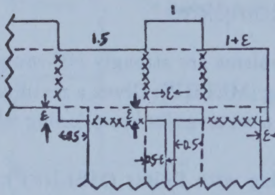


Figure 11. A wire-split; the two decompositions show a '0' and a '1'.

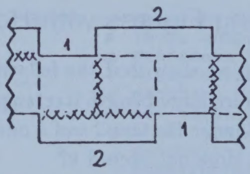


Figure 12. An inverter: a '0' (perpendicular decomposition) on the left turns into a '1' (parallel decomposition) the right and vice versa.

to a unique 90^0 turn in a wire of $\mathcal{H}$, called a *variable turn*; each vertex in C corresponds uniquely to a junction of three wires in $\mathcal{H}$, called a *clause junction*. Each edge in E is realized by a wire path between the corresponding variable turn and the clause junction. A wire is supposed to enter the clause junction carrying the truth value of the literal it corresponds to; since no distinction between x_i and $\bar{x}_i$ was made in constructing $G(\mathcal{F})$, we need to install *inverters* on the entry to a junction, as described in Figure 12. Also, wires have to enter turns and junctions with an appropriate phase for their sides, thus a *phase shifter* may be applied, as shown in Figure 13. Figure 14 shows a clause junction. Note that the pair of points P_3, P_6 is slightly not corectilinear (by ϵ); so is the pair P_3, P_{10}. Therefore, the wire entering at the bottom is narrower by $\epsilon < \frac{1}{4 \cdot \nu(\mathcal{H})}$, where $\nu(\phi)$ denotes the number of concave corners in the figure ϕ. We shall see that clause junctions behave like three-input *OR* gates.

Let us now cut $\mathcal{H}$ into single wires, wire-splits, inverters, phase shifters and junctions (such cuts are denoted in our pictures by ragged lines). Let $\mathcal{D}(\mathcal{H})$ denote the set of *devices* into which $\mathcal{H}$ has been cut. Given a partitioning $\mathcal{R}$ of $\mathcal{H}$ into rectangles, we associate with each edge $e \in \mathcal{R}$ the value $s_{\mathcal{R}}(e)$ to be equal to the length of e divided by the number of devices in $\mathcal{D}(\mathcal{H})$ that contain a continuous piece of e. For every $d \in \mathcal{D}(\mathcal{H})$, $\text{charge}_{\mathcal{R}}(d)$ is the sum of $s_{\mathcal{R}}(e)$ over all edges $e \in \mathcal{R}$ that have a continuous piece in d. Clearly,

$$\sum_{e \in \mathcal{R}} \text{length}(e) = \sum_{d \in \mathcal{D}(\mathcal{H})} \text{charge}_{\mathcal{R}}(d).$$

We say that a device d is *optimally charged* by a partitioning $\mathcal{R}$ of $\mathcal{H}$ if for any other partitioning, $\mathcal{Q}$,

$$\text{charge}_{\mathcal{R}}(d) - \text{charge}_{\mathcal{Q}}(d) < \epsilon \cdot \nu(d).$$

Then we have

Lemma 5. For any wire, wire-split, inverter and phase shifter d, d is optimally charged

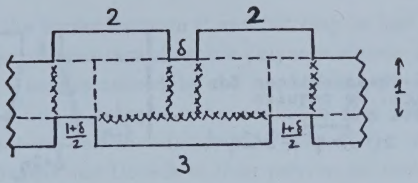
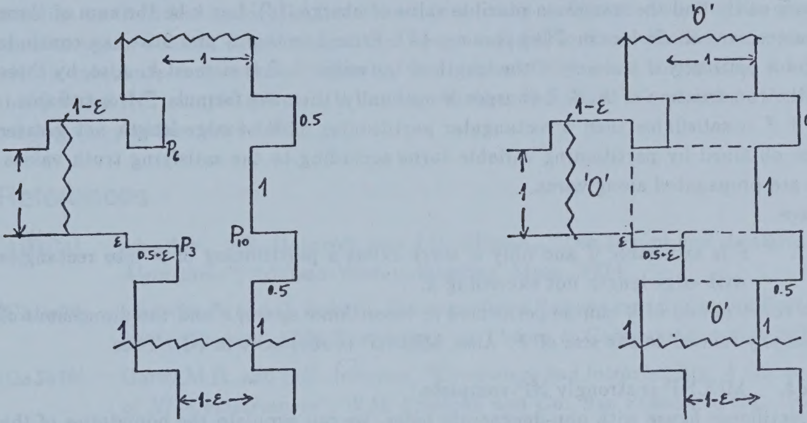
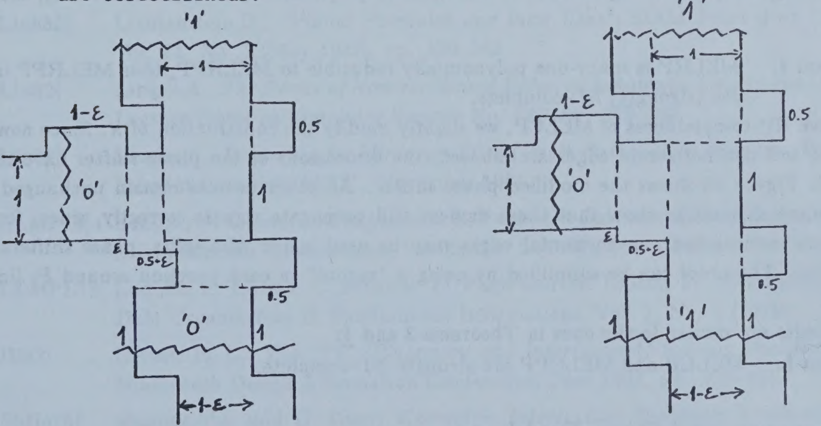

2
2
δ
1
$\frac{1+\delta}{2}$
$\frac{1+\delta}{2}$
3

Figure 13. A phase shifter. The phase is shifted by δ.


1
1-ε
0.5
P_6
1
1
ε
0.5+ε
P_3
P_{10}
0.5
1
1
1-ε
'0'
1
1-ε
0.5
'0'
1
1
ε
0.5+ε
0.5
'0'
1
1
1-ε

(a) Layout of the junction. Neither P_3, P_6 nor P_3, P_{10} are corectilinear.

(b) All '0' inputs yield a decomposition of length 5.5


'1'
1
1-ε
0.5
'0'
1
1
ε
0.5+ε
0.5
'0'
1
1
1-ε
'1'
1
1-ε
0.5
'0'
1
1
ε
0.5+ε
0.5
'1'
1
1
1-ε

(c) One '1' is enough to yield a decomposition of length 5+ε.

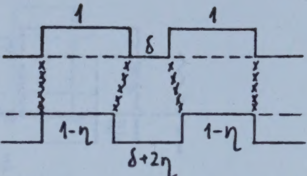
(d) Again, the decomposition length is 5+ε.

Figure 14. A clause junction and how it is being (negatively) charged.

by $\mathcal{R}$ if and only if d propagates the signal correctly. Otherwise, the charge is greater by at least $\frac{1}{2} - \epsilon \cdot \nu(d)$.

Lemma 6. If at least one wire entering a clause junction d carries a 1, then d can be charged optimally by $\mathcal{R}$ (of that wire). Otherwise, $\text{charge}_{\mathcal{R}}(d)$ is greater by at least $\frac{1}{2} - 2 \cdot \epsilon$.

Figure 15. A δ-phase-shifter for MELCD. We require $0<\delta\leq 1$ and $2\cdot(1+\sqrt{1+\eta^2})=3+2\eta+\delta$.



We say that $\mathcal{R}$ charges $\mathcal{H}$ optimally if every $d \in \mathcal{D}(\mathcal{H})$ is charged optimally by it. For each device d we can easily find the minimum possible value of $\text{charge}_{\mathcal{R}}(d)$. Let k be the sum of these minimum charges over all devices in $\mathcal{D}(\mathcal{H})$ plus $\epsilon \cdot \nu(\mathcal{H})$. From Lemmas 5 and 6 we can conclude that $\mathcal{R}$ charges $\mathcal{H}$ optimally if and only if the length of the edges in $\mathcal{R}$ is at most k. Also, by these lemmas and the construction of $\mathcal{H}$, if $\mathcal{R}$ charges $\mathcal{H}$ optimally, then the formula $\mathcal{F}$ is satisfiable.

Finally, if $\mathcal{F}$ is satisfiable then a rectangular partitioning of $\mathcal{H}$ of edge length not greater than k can be obtained by partitioning variable turns according to the satisfying truth values. Then signals are propagated along wires.

So we have

Lemma 7. $\mathcal{F}$ is satisfiable if and only if there exists a partitioning of $\mathcal{H}$ into rectangles with edge length not exceeding k.

Since the construction of $\mathcal{H}$ can be performed in logarithmic space, k and the dimensions of $\mathcal{H}$ are polynomially related to the size of $\mathcal{F}$. Also, MELRP is obviously in NP. Thus

Theorem 3. MELRP is strongly NP-complete.

Given a rectilinear figure with non-degenerate holes, we can simulate the boundaries of the holes by appropriately dense (corectilinear) points. Together with the fact that we can find an optimal rectangular partitioning of each of the holes in polynomial time (by Theorem 1), this leads to

Theorem 4. MELRP is many-one polynomially reducible to MELRPP; thus MELRPP is also (strongly) NP-complete.

To prove NP-completeness of MELCP, we slightly modify the construction of $\mathcal{H}$. Since now non-vertical and non-horizontal edges are allowed, the dimensions of the phase shifter have to be changed. Figure 15 shows the modified phase shifter. All other devices remain unchanged. Now it is more difficult to show that these devices still propagate signals correctly, since, for instance, non-vertical and non-horizontal edges may be used inside wire-splits, phase shifters, and junctions. The proof can be simplified by using a "sprout" in each junction around P_3 (in Figure 14).

The results are similar to the ones in Theorems 3 and 4:

Theorem 5. MELCP and MELCPP are strongly NP-complete.

5. Extensions and Open Problems

We have shown that the complexity of partitioning a rectilinear figure into rectangles using minimum edge length depends on whether holes are permitted or not. In general the problem is NP-complete, but when holes are disallowed the problem becomes tractable. Two major open problems remain unsolved:

(i) What if all holes are degenerate (*i.e.* they are points), but no two points are corectilinear (*i.e.* lie on the same horizontal or vertical line)?

(ii) What if, as in (i), no two points are corectilinear, and the boundary is a rectangle?

We could show neither of the problems to be NP-complete or polynomial-time solvable. Clearly, (ii) is "easier" than (i), so that resolving one in the right direction may solve them both, but on the other hand the border between P and NP may lie just between these two problems.

Other interesting extensions regarding the polynomial-time algorithm of Section 3 are the following: First, can it be generalized to the non-rectilinear case? That is, can we find a *convex* partitioning of a simple polygon using minimum edge length in polynomial time? How about triangulations that use non-boundary points? There is an interesting balance between the convexivity that led Chazelle and Dobkin to their polynomial-time algorithm for minimizing the *number* of pieces on one hand, and the Steiner tree-like property of our new minimum *edge length* criterion on the other hand. The latter took over in the rectilinear case, but the general problem remains open. Second, can the algorithm be generalized to three dimensions, where the metric will be minimum surface area? Clearly, the results of Section 4 imply that solid holes make this problem NP-complete, but it is unclear whether the added dimension in the hole-free case renders the problem intractable.

References

[AHU74] Aho,A.V., J.E. Hopcroft and J.D. Ullman, *"The Design and Analysis of Computer Algorithms"*; Addison-Wesley, Reading, Mass., 1974.

[ChDo79] Chazelle,B. and D. Dobkin, *Decomposing a Polygon into its Convex Parts*; Proceedings of the Eleventh ACM Symposium on Theory of Computation, May 1979, pp. 38-48.

[GaJh79] Garey,M.R. and D.S. Johnson, *"Computers and Intractability: A Guide to the Theory of NP-Completeness"*; W.H. Freeman and Co., San Francisco, Ca., 1979.

[GJPT78] Garey,M.R., D.S. Johnson, F.P. Preparata and R.E. Tarjan, *Triangulating a Simple Polygon*; Information Processing Letters, Vol. 7, No. 4 (June 1978), pp. 175-180.

[Lich82] Lichtenstein,D., *"Planar Formulae and their Uses"*; SIAM Journal on Computing, Vol. 11, No. 2 (May 1982), pp. 329–343.

[Lin82] Lingas,A., *The Power of Non-rectilinear Holes*; in Automata, Languages and Programming, Lecture Notes in Computer Science No. 140, pp. 369–383.

[Lin81] Lingas,A., *Heuristics for Minimum Edge Length Rectangular Decomposition*; unpublished manuscript, M.I.T., November 1981.

[LLMP79] Lodi,E., F. Luccio, C. Mugnai and L. Pagli, *On Two-dimensional Data Organization I*; Fundamenta Informaticae, Vol. 2, No. 2 (1979), pp. 211-226.

[LLMPL79] Lodi,E., F. Luccio, C. Mugnai, L. Pagli and W. Lipski, Jr., *On Two-dimensional Data Organization II*; Fundamenta Informaticae, Vol. 2, No. 3 (1979).

[Ri82] Rivest, R. L.: *The "PI" (Placement and Interconnect) System*; Proceedings of the Nineteenth Design Automation Conference, June 1982, pp. 475–481.

[ShHo76] Shamos,M.I. and D. Hoey, *Geometric Intersection Problems*; Proceedings of the Seventeenth IEEE Symposium on Foundations of Computer Science, October 1976, pp. 208-215.

An O(nlogn) Algorithm for
Decomposing Simple Rectilinear Polygons into Convex Quadrilaterals

Jörg-R. Sack

It has been shown that rectilinear polygons always admit a decomposition into convex quadrilaterals. Recently an optimal algorithm for quadrilaterization of star-shaped rectilinear polygons has been presented. Here these results are extended to simple, rectilinear, n-vertex polygons. This paper contains three algorithms. The first algorithm tests, in linear time, whether an arbitrary rectilinear polygon is monotone. The second algorithm decomposes, in linear time, monotone rectilinear polygons into convex quadrilaterals. The third algorithm decomposes simple rectilinear polygons, in O(nlogn) time, into special monotone polygons admitting a convex quadrilaterization. The complexity to decompose a simple rectilinear polygon into convex quadrilaterals is therefore O(nlogn).

Author's Address: School of Computer Science
McGill University
805 Sherbrooke St. West
Montreal, Quebec H3A 2K6
Canada

I. INTRODUCTION

Certain shapes of objects occurring in image analysis are efficiently described by specific classes of polygons, such as convex, monotone, or star-shaped polygons. Algorithms working on these classes of polygons very often have smaller time and/or space complexities. For example, star-shaped, monotone, and edge-visible polygons can be triangulated in O(n) time [4,16,8,19], whereas the best algorithm presently known for triangulating a simple polygon takes O(nlogn) time [8]. Techniques to decompose objects into simpler components, decomposition techniques, have received widespread attention in pattern recognition [6,11], image analysis, and computational geometry. O(nlogn) algorithms for decomposing simple polygons into star-shaped polygons and into monotone polygons have been presented in [1, 8]. Decomposition into convex parts has been studied in [3,15]. For a survey of polygon decomposition methods see [18].

Definition: A rectilinear polygon $P = p_1, p_2, \ldots, p_n$ is a simple polygon in which all the edges are parallel to two given orthogonal directions. The vertices are given in clockwise order. Without loss of generality, in this paper we consider the x and y axes as the two directions since an arbitrary n-vertex polygon can be rotated in O(n) time. In image processing rectilinear polygons are frequently used to efficiently describe images on rectangular grids [7,17].

Definition: A point x in P sees a point y in P, if the open line segment joining x and y lies entirely inside P.

Definition: A polygon P is star-shaped if there exists at least one point x inside P such that x can see the entire polygon. The set of all points x in P such that x can see the entire polygon is called the kernel of P.

Definition: A polygon P is weakly visible from an edge pq of P if for every point x in P there exists a point y in pq that sees x. P is edge-visible if there exists at least one edge from which the polygon is weakly visible.

Definition: *A point* x *sees an edge* e, if there exists a point y on e such that x sees y.

Definition: *An edge* s *sees an edge* t, if there exist points x on s and y on t, such that x sees y. Clearly point to point and edge to edge visibility are symmetrical.

Definition: A *diagonal* is a line segment joining two nonconsecutive polygonal vertices which lies entirely in P.

Definition: A *chain* C_{ij} is a sequence of consecutive vertices $p_i, \ldots, p_j$ of P. A chain is *monotone* with respect to a line l if the projections of p_k, $k=i,\ldots,j$, on l are ordered in exactly the same way as the vertices in C_{ij}.

A decomposition of a polygon into convex quadrilaterals will be called a *quadrilaterization*. Quadrilaterization and *triangulation*, the decomposition of polygons into triangles, are the most common decomposition methods, if decomposition into fixed "shaped primitives" [11] is of interest. As mentioned above a simple polygon can be triangulated in O(nlogn) time, whereas a simple polygon does not always admit a quadrilaterization. Kahn, Klawe, and Kleitman proved that a quadrilaterization always exists if the polygon is rectilinear [9]. We give an efficient algorithm for this decomposition problem and we analyze the complexity of the algorithm. An O(n), and thus optimal, algorithm for decomposing rectilinear star-shaped polygons into convex quadrilaterals has been presented in [13]. In this paper we extend these results by first finding a quadrilaterization for monotone rectilinear polygons. Then we show how to decompose a simple rectilinear polygon into special monotone polygons that admit a quadrilaterization. Additionally we prove some important results about rectilinear polygons, including a characterization of monotone rectilinear polygons. Let us briefly review some of the definitions and lemmas of the paper by Kahn, Klawe, and Kleitman.

Definition: An edge e of a polygon P is called a *top*, *bottom*, *left*, or *right* edge, if the interior of P is below, above, to the right, or to the left of e respectively. Recall that the vertices of P are given in clockwise order.

Definition: A top edge s and a bottom edge t which see each other are called *neighbours*, if s is higher than t, and there is no bottom (top) edge visible to s (t) whose vertical coordinate lies between those of s and t. A *tab* is a pair of neighbouring edges which are connected to each other by a vertical edge. A tab is a *down-tab* if its top edge extents further inside P than its bottom edge; otherwise the tab is called an *up-tab*.

If a is a point or a vertical edge and b is a point or a horizontal edge, then we will use a#b to denote the point whose horizontal coordinate is that of a, and whose vertical coordinate is that of b. Similarly for b#a (see figure 1).

Lemma 1: If $p_{i-1}, p_i, p_{i+1}, p_{i+2}$ is a tab, then any quadrilaterization contains the diagonal $p_{i-1}p_{i+2}$ (see figure 2).

Lemma 2: Let P be a rectilinear polygon containing two neighbouring edges $s = p_{i-1}p_i$ and $t = p_jp_{j+1}$, which do not form a tab. Then P can be decomposed in two polygons $P_1 = p_i, \ldots, p_j$ and $P_2 = p_{j+1}, \ldots, p_{i-1}$, which both admit quadrilaterization. p_ip_j and $p_{j+1}p_{i-1}$ are the diagonals inserted.

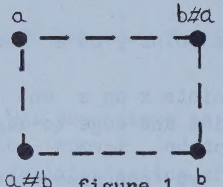


figure 1

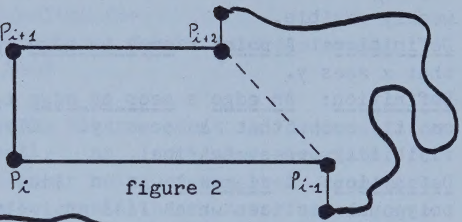


figure 2

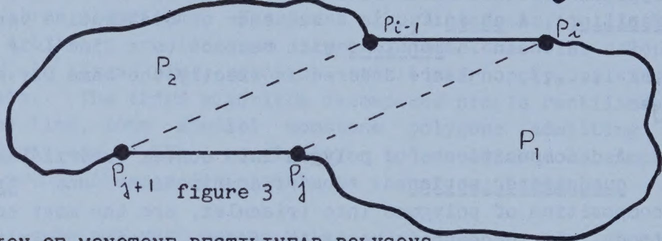


figure 3

II. CHARACTERIZATION OF MONOTONE RECTILINEAR POLYGONS

In this paragraph we give a characterization of monotone rectilinear polygons. We prove a theorem for monotone rectilinear polygons that enables us to describe an algorithm for testing whether a simple rectilinear polygon is monotone. The existing algorithm for testing whether an arbitrary simple polygon is monotone, by Preparata and Supowit [12], has the same worst case complexity. The algorithm proposed here, however, is much simpler, and thus has a smaller multiplicative constant, because we are able to exploit the structure of rectilinear polygons.

Let l be any line, the orientation of l will be described by the counterclockwise angle w formed by l and the x axis (see figure 4). We say that the monotonicity of a polygonal chain C_{ik}= $p_i,\dots,p_j$, ..., p_k with respect to a given line l is changed at $p_j p_{j+1}$ if the order of the projections of $p_{j-1}, p_j, p_{j+1}, p_{j+2}$ onto l differ from the polygonal order.

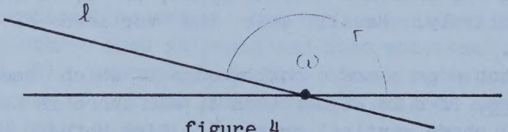


figure 4

Lemma 1: If a rectilinear polygon P is monotone with respect to a line l, with w in (0, /2), then P is monotone with respect to any line l with w in [0, /2]. Similarly for w in (/2,).

Proof: Let P be monotone with respect to any line l, with w in (/2,), see figure 5. The case where w is in (0, /2) can be handled similarly. Any rectilinear polygon P contains edges y_{max}, y_{min} with maximum, minimum y-value respectively. The monotonicity with respect to l is changed at the edges y_{max} and y_{min}. W.l.o.g. let $y_{max} = p_n p_1$ and $y_{min} = p_k p_{k+1}$. If P is monotone with respect to l, then both chains have to be monotone with respect to l. The chains cannot contains tabs, because tabs change the monotonicity. Therefore, as P is monotone with respect to l, all horizontal edges on the left chain are bottom edges and all horizontal edges on the right chain are top-edges (see figure 6).

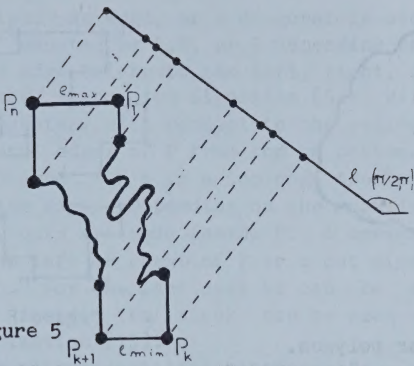


figure 5

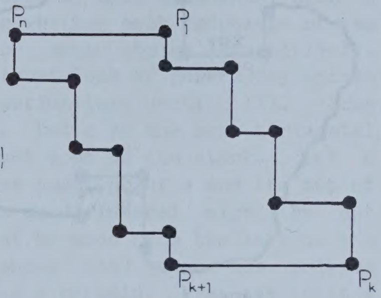


figure 6

But these polygons, which we will call *down-escalator*, are clearly monotone with respect to any line l with w in $[\pi/2, \pi]$, e.g. they are monotone to the x and the y-axes. The proof for the case when P is monotone with respect to a line l with w in $(0, \pi/2)$ is similar (rotate figure 6 by $\pi/2$ to get this type of polygons). These polygons will be called *up-escalator*. Incidentally the rectilinear polygon with only four vertices, the rectangle, is the only rectilinear polygon monotone with respect to all lines l with w in $[0, \pi]$.

We can conclude that any rectilinear polygon P monotone with respect to some line l is monotone with respect to the x-axis or the y-axis. Furthermore, if P is monotone to any line not parallel to one of the two major axes, then P is monotone to all lines l with w in $[0, \pi]$ or w in $[\pi/2, \pi]$. Q.E.D. A rectilinear polygon monotone with respect to the y-axis is illustrated in figure 8, where $p_1, \ldots, p_k$ is the left chain and $p_{k+1}, \ldots, p_n$ is the right chain.

We can restrict ourselves therefore to monotonicity with respect to any one of the two major axes, say the y-axis, knowing that any rectilinear polygon can be rotated in O(n) time.

Let all vertical edges be directed from top to bottom, although this is strictly speaking a directed graph, we use the term polygon. Given a rectilinear polygon P and a horizontal edge e we can denote the number of incoming (outgoing) edges into (out off) e by In(e), (Out(e)) respectively. As P is a simple polygon In(e) + Out(e) = 2, for all horizontal edges e in P.

Definition: A bottom-edge e is called a *bottom-peak*, if Out(e) = 2. A top-edge is called a *top-peak*, if In(e) = 2, see figure 7. Similarly we could define *left peaks* and *right peaks*. Top and bottom peaks are called *horizontal peaks*, left and right peaks are called *vertical peaks*.

Definition: A *pyramid* is defined as a rectilinear edge-visible star-shaped polygon . (This is a slightly more general definition than in [13], because rotation is allowed.)

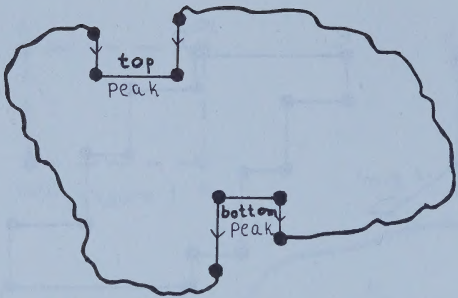


figure 7

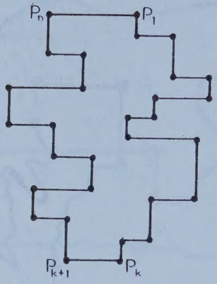


figure 8

Theorem: Let P be a rectilinear polygon.

(i) P does not contain any horizontal peaks if and only if P is monotone with respect to the y-direction.

(ii) P does not contain any vertical peaks if and only if P is monotone with respect to the x-direction.

(iii) A rectilinear polygon P does not contain any peaks if and only if
(a) P is an escalator,
or (b) P is monotone with respect to the x-axis and the y-axis and P is not monotone with respect to any other line.

Remark: A vertex v of a simple polygon is called an interior cusp if the internal angle at v exceeds 180 degrees and the two vertices adjacent to q on the boundary of P either have both larger y-coordinates than q or both have smaller y-coordinates than q. The "only-if"-direction in (i) is similar to Gary, Johnson, Preparata, Tarjan's theorem 1: If P is a simple polygon and no vertex of P is an interior cusp, then P is monotone [8].

As a consequence of the theorem we can obtain an O(n) algorithm for testing whether a rectilinear polygon is monotone: In O(n)-time we can test whether P contains any peaks. If P does not contain any horizontal peaks, then P is monotone in the y-direction. If P does not contain any vertical peaks, then P is monotone in the x-direction. If P is monotone in the x-direction and P is montone in the y-direction, then the algorithm tests in constant time whether P is an escalator:

If the y_{max}-edge is adjacent to the x_{min}-edge and the y_{min}-edge is adjacent to the x_{max}-edge then P is monotone to all lines l with w in [$\pi/2,\pi$] and therefore P is an down-escalator.

If the y_{max}-edge is adjacent to the x_{max}-edge and the y_{min}-edge is adjacent to the x_{min}-edge then P is monotone to all lines l with w in [$0,\pi/2$]; and therefore P if an up-escalator.

In all other cases P is monotone with respect to the x-axis and to the y-axis, but not monotone with respect to any other line. Another constant time test could determine if P is a pyramid or a star-shaped polygon.

III. Decomposing monotone rectilinear polygons into convex quadrilaterals

In this section we will show how to find a convex quadrilaterization of a monotone rectilinear polygon in linear time.

Data Structure: We maintain a stack S containing some horizontal edges and some diagonals. Inside the stack S the edges and diagonals are sorted by y-coordinates. An edge, or a diagonal is stored by both endpoints, plus a fieldvalue denoted by L,R, or B depending on whether both endpoints of the diagonal or edge belong to the left, right, or both chains respectively. We will now state the algorithm [5]: Without loss of generality assume that P is monotone with respect to the y-direction (see section II). Scan the horizontal edges of P from top to bottom. Let e be the next horizontal edge encountered. If e is a top edge then push e on to the stack. Let e be a bottom edge. Depending on the relative position of e and the top of stack edge, cuts are made inside P. A convex quadrilateral might be cut off from the left or right of P or a cut might be made from the left to the right chain. For the last case it can be shown [14] that the polygon maintained inside the stack can be seen as a pyramid. Pyramids admit a quadrilaterization [13].

Input: A monotone rectilinear polygon P.

Output: A quadrilaterization of P.

```
Algorithm II:
sort, in linear time, the horizontal edges in order of decreasing y-values;
let $e_1$, ..., $e_n$ be the sorted edges;
Initialize the stack S <-- [$e_1$,B];
i <-- 1;
While i<> n or S<> empty do
  begin
(a) if edge $e_i$ is a top edge then
      begin
        S<-- [$e_i$, respective fieldvalue];
        i <-- i+1
      end
    else (* $e_i$ is a bottom edge *)
      begin
        e <-- POP(S);
          if e is a diagonal with fieldvalue B
          or e and $e_i$ are on the same chain then
(b)       begin  (* e and $e_i$ can be seen as edges forming a tab *)
            join e and $e_i$ by a diagonal;
            if e is a diagonal with fieldvalue B
            or e is further inside P than $e_i$ then
              begin
                S <-- the diagonal with respective fieldvalue;
                i <-- i + 1
              end
          end
(c)     else (* e, $e_i$ are on opposite chains *)
          begin
            join e and $e_i$ to form a quadrilateral;
            S <-- left diagonal;
            quadrilaterize the pyramid contained in S;
            i <-- i+1;
            S <-- right diagonal with fieldvalue B
          end
      end  (* edge $e_i$ is a bottom edge  *)
  end  (* end for while loop *)
end of algorithm II.
```

Theorem: Monotone rectilinear polygons are quadrilateriable in linear time.
The proof for this theorem can be found in [14].

IV. Decomposing_Rectilinear_Polygons_into_Monotone_Polygons

In this section we describe an algorithm to decompose a rectilinear n-vertex polygon into special monotone polygons admitting a quadrilaterization. The run-time of the algorithm will be shown to be O(nlogn).

Definition: An edge e is visible from a point x if there exists a point y on e such that x sees y.

Definition: An edge e is strongly_visible from a point x if point x sees all points on e.

Definition: An edge s is strongly_visible_from_an_edge t if there exists a point x on t from which s is strongly visible. Strong visibility is not necessarily symetrical.

Definition: An edge s is completely_visible from an edge t if for all x on t the edge s is strongly visible from x. Complete visibility is symmetrical.

Lemma_1: Let qp be any edge of a simple polygon, and let x be a point in P. If both q and p see x, then x is visible from the entire edge qp.

Proof: D. Avis and G.T. Toussaint proved that if q and p are visible from x, then so is the entire edge qp [2]. As visibility between two points is symmetrical the lemma follows.

Lemma_2: Let qp be any bottom-peak in a rectilinear polygon P. Let ab be the horizontal edge located above qp with the minimum distance in y-coordinate that is visible from q or p, then

(i) The edge qp is strongly visible from ab.
(ii) If ab is a top-edge then ab and qp are completely visible from each other.

Proof: Using lemma 1 we only have to show that:

(a) If a point x on ab is visible from q or p then x is visible from q and p.
(b) If ab is a top-edge visible from q, then a and b are visible from q.

(a) W.l.o.g. let x be visible from q. Now consider the triangle T = {x,p,q} and refer to figure 9. If T lies entirely inside P then p sees x. Otherwise px is intersected by a polygonal edge and as qp is a bottom-peak this edge is not adjacent to qp. Therefore there exists a vertex p_i inside T. As P is simple there exists a horizontal edge with y-coordinate between qp and ab intersecting T, the lowest such edge is visible from q, contradiciting the choice of ab.

(b) Similar to (a).

We conclude that if ab is a bottom- edge, then the edge qp is strongly visible from exactly one point on ab, either the point a or the point b. Furthermore, if ab is a top-edge, then the edges ab and qp are neighbours and are thus completely visible from each other.

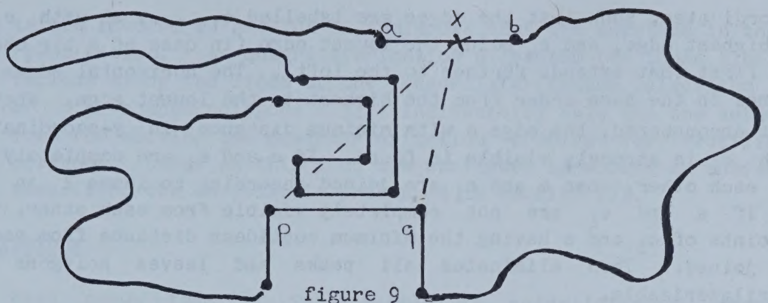


figure 9

Remark: Let the assumptions be as in lemma 2. Then, if ab is a bottom-edge, ab lies either to the right of q or to the left of p (see figure 10). The proof follows along the same lines as the lemma and is omitted.

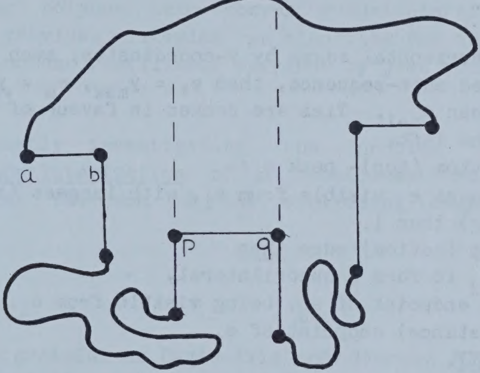


figure 10

Corollary: Given the assumptions of lemma 2. The rectangles (p#a,a,a#q,p) shown in figures 11 (a) and 11 (b) are empty.

Proof: If ab is a bottom-edge, refering to figure 11(a), qp is strongly visible from one endpoint of ab, say from a. Thus the triangle T = {a,q,p} is empty. The two adjacent triangles T_1 = {a,p,p#a} and T2 = {a,q,a#q} are likewise empty, otherwise a horizontal edge between ab and qp visible from p or q could be found. Similarly, if ab is a top-edge, then ab and pq are completely visible from each other, (see lemma 2), and the rectangle depicted in figure 11a is empty.

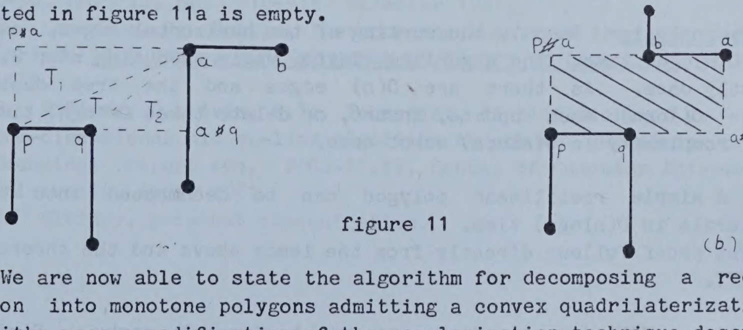


figure 11

We are now able to state the algorithm for decomposing a rectilinear polygon into monotone polygons admitting a convex quadrilaterization. The algorithm uses a modification of the regularization technique described by Lee and Preparata [10]. Let the horizontal edges be sorted by y-coordinates, such that the edges are labelled $e_1, \ldots, e_m$ with e_1 being the highest edge, and e_m being the lowest edge (in case of a tie choose the edge first that extends further to the left). The horizontal edges e_i are scanned in the same order from the highest to the lowest edge. When a peak e_i is encountered, the edge e with minimum distance in y-coordinate from which e_i is strongly visible is found. If e and e_i are completely visible from each other, then e and e_i are joined according to lemma 2 in section I. If e and e_i are not completely visible from each other, then the endpoints of e_i and e having the minimum euclidean distance from each other are joined. This eliminates all peaks and leaves polygons that are quadrilaterizable.

Input: A rectilinear polygon P with n vertices.
Output: A decomposition into monotone polygons admitting a convex quadrilaterization.

```
Algorithm III:
Step 1: Sort the horizontal edges by y-coordinates, such that if $e_1,\ldots,e_m$
        is the ordered edge-sequence, then $e_1 = y_{max}$, $e_m = y_{min}$, and $e_i$ is
        not lower than $e_{i+1}$. Ties are broken in favour of the edge reaching
        further to the left.
repeat for each bottom (top)- peak $e_i$:
Step 2: Find the edge $e_j$ visible from $e_i$, with largest (smallest) index j
        less (greater) than i.
      If $e_j$ is a top (bottom) edge then
      join $e_i$ and $e_j$ to form a quadrilateral,
      else join the endpoint of $e_j$, being visible from $e_i$, to the closest
      (euclidean distance) endpoint of $e_i$.
end of algorithm III.
```

For each horizonal peak e (in step 2) an AVL-tree can be used to find the closest horizontal edge visible from e. The details can be found in [14]. This is a modification of Lee's and Preparata's [10] regularization-technique, in which horizontal edges instead of vertices are inserted into the AVL-tree, and only one path through the polygon is necessary.

Lemma: Algorithm III decomposes, in O(nlogn) time, a simple rectilinear polygon into monotone polygons admitting quadrilaterization.

Complexity-analysis: Step 1, the sorting of the horizontal edges, can be done in O(nlogn) time. The algorithm visits, while executing step 2, each edge exactly once. As there are O(n) edges and the tree depth is worst-case O(logn) each update, insert, or delete takes O(logn) time and the total complexity is O(nlogn) worst case.

Theorem: A simple rectilinear polygon can be decomposed into convex quadrilaterals in O(nlogn) time.
Proof: The proof follows directly from the lemma above and the theorem in section III.

Algorithm III inserts diagonals at horizontal peaks. Similarly diagonals at vertical peaks could be inserted. This process would yield, as shown in the theorem in section II, polygons that are monotone in the x and the y-direction. A quadrilaterization algorithm for these types of polygons follows along the lines of the quadrilaterization algorithm for rectilinear star-shaped polygons [13]. Incidentally this was the authors' original idea for solving the quadrilaterization problem for rectilinear monotone polygons. Later on this idea was abolished to obtain an easier to describe algorithm, the one that is presented in section III.

V. Conclusions

We have shown that it is possible to quadrilaterize monotone rectilinear polygons in linear time and to decompose, in O(nlogn) time, rectilinear polygons into monotone polygons admitting convex quadrilaterization. The total complexity for decomposing an arbitrary simple rectilinear polygon into convex quadrilaterals is therefore O(nlogn). This provides likewise an algorithm for triangulating simple rectilinear and monotone rectilinear polygons by just adding a diagonal to each quadrilateral.

We are presently investigating the problem of finding the minimum-weight-quadrilaterization of a rectilinear polygon, where the weight is defined as the sum of the euclidean lengths of the added diagonals.

Acknowledgement

The author is grateful to David Avis and Hossam ElGindy for useful discussions on this topic, likewise to G.T. Toussaint and Mike McKenna for reviewing the manuscript.

References

[1] D. Avis, G.T. Toussaint, "An efficient algorithm for decomposing a polygon into star-shaped polygons", Pattern Recognition, Vol. 13, pp. 395-398, 1981. 1981.

[2] D. Avis, G.T. Toussaint, "An optimal alogorithm for determining the visibility of a polygon from an edge", IEEE Transactions on Computers, C-30, No. 12, pp. 910-914, December 1981.

[3] B. Chazelle, D. Dobkin, "Decomposing a polygon into its convex parts", 11th annual ACM Symposium on Theory of Computing, pp. 38-48, 1979.

[4] H. ElGindy, D. Avis, G.T. Toussaint, "Applications of a two-dimensional hidden-line algorithm to other geometric problems", Technical Report No. SOCS-81.13, School of Computer Science, McGill University, April 1981.

[5] H. ElGindy, personal communication.

[6] H. Feng, T. Pavlidis, "Decomposition of polygons into simpler components: feature generation for syntactical pattern recognition", *IEEE Transactions on Computers*, C-24, No. 6, pp. 636-650, June 1975.

[7] L. Ferrari, P.V. Sonka, J. Sklansky, "Minimal rectangular partitions of digitized blobs", *Proc. Fifth International Conf. on Pattern Recognition*, Vol 2, Miami Beach, pp. 1040-1043, Dec 1980.

[8] M.R. Garey, D.S. Johnson, F.P. Preparata, and R.E. Tarjan, "Triangulating a simple polygon", *Information Processing Letters*, Vol 7, pp. 175-179, June 1978.

[9] J.Kahn, M.Klawe, D. Kleitman, "Traditional galleries require fewer watchmen", RJ3021 (37386) 12/22/80, Computer Science Department, IBM, San Jose, California 95193, Mathematics Department, Massachusetts Institute of Technology.

[10] D.T.Lee, F.P.Preparata, "Location of a point in a planar subdivision and its application", *SIAM Journal on Computing*, Vol. 6, pp. 594-606, 1977.

[11] T. Pavlidis, "Structural pattern recognition", Springer Series in Electrophysics Vol 1, Springer Verlag Berlin, Heidelberg, New York, 1977.

[12] F.P. Preparata, K.J. Supowit, "Testing a simple polygon for monotonicity", *Information Processing Letters*, Vol. 12, pp. 161-164, 1981.

[13] J.R. Sack, G.T. Toussaint, "A linear-time algorithm for decomposing rectilinear star-shaped polygons into convex quadrilaterals", *Proceedings of The Allerton Conference*, Urbana, Illinois, Oct. 1981.

[14] J.R. Sack "An O(nlogn) algorithm for decomposing simple rectilinear polygons into convex quadrilaterals", unpublished manuscript.

[15] B. Schachter, "Decomposition of polygons into convex sets", *IEEE Transactions on Computers*, C-27, No. 11, pp. 1078-1082, Nov 1978.

[16] A.A. Schoone, J. van Leeuwen, "Triangulating a star-shaped polygon", University of Utrecht, RUU-CS-80-3, April 1980.

[17] J. Sklansky, "Recognition of convex blobs", *Pattern Recognition*, Vol 2, pp. 3-10, 1970.

[18] G.T. Toussaint, "Pattern recognition and geometrical complexity", *Proc. Fifth International Conf. on Pattern Recognition*, Vol 2, Miami Beach, pp.1324-1347, Dec. 1980.

[19] G.T. Toussaint, D. Avis, "On a convex hull algorithm for polygons and its applications to triangulation problems", *Pattern Recognition*, Vol. 15, pp. 23-29, 1982.

THE COMPLEXITY OF COMPUTING MINIMUM CONVEX COVERS FOR POLYGONS

JOSEPH O'ROURKE
Department of Electrical Engineering and Computer Science
The Johns Hopkins University
Baltimore, MD 21218

ABSTRACT

The computational complexity of the problem of finding a minimum convex cover for a polygon is reviewed. Although an NP-hard lower bound has been proven, no upper bound on the complexity is known. The lack of known restrictions on the Steiner points that may be necessary prevents naive searching algorithms from establishing the solvability of the problem. In this paper the problem is proven to be decidable by using Tarski's decision procedure for geometry.

1. Introduction

Given an arbitrary polygonal region of the plane, perhaps with polygonal holes, it is frequently useful to decompose the region into convex polygons. Such decompositions find application in pattern recognition [17-20] [23], artificial intelligence and graphics [15], and VLSI design [6]. If the convex polygons used in the decomposition are permitted to overlap, the decomposition is called a *convex cover*. The problem discussed in this paper is the computational complexity of finding the *minimum* convex cover, minimum in the sense of having the fewest number of pieces.

The problem formulation in the style of Garey and Johnson [5] is as follows [6]:

COVERING BY CONVEX POLYGONS (CCP)

INSTANCE: A polygon P (not necessarily simply-connected), described by a sequence of n pairs of integer coordinate points in the plane, each pair representing the endpoints of a line segment on the boundary of P, and a positive integer K.

QUESTION: Are there convex polygons C_i, $1 \leq i \leq K$, whose union (as planar figures) is P ?

Decomposing a polygonal region into *non*-overlapping pieces (finding a *partition*) is a closely related but distinct problem. Minimum partitions of both simply-connected [3] [2] and multiply-connected regions can be found in polynomial time. These results unfortunately do not bear directly on minimum covers.

The use of Steiner points is discussed in the next section, and Section 3 shows that naive algorithms could miss finding the true minimum cover. Section 4 summarizes a proof of NP-hardness for the problem, establishing a lower bound, and the section following proves that the problem is decidable (a new result), establishing an upper bound. Open problems are formulated in the final section.

2. Steiner Points and Naive Algorithms

It is instructive to consider brute-force searching algorithms for finding minimum convex covers: such algorithms at least establish that the problem they solve is *decidable*. In general, *Steiner points*, points that are not vertices of the polygon being decomposed, may be required as vertices of the convex pieces of a minimum cover. If the problem is restricted to find minimum covers without the use of Steiner points, then a searching algorithm is possible: con-

---

*This research was supported by the National Science Foundation under Grant MCS 81-17424.

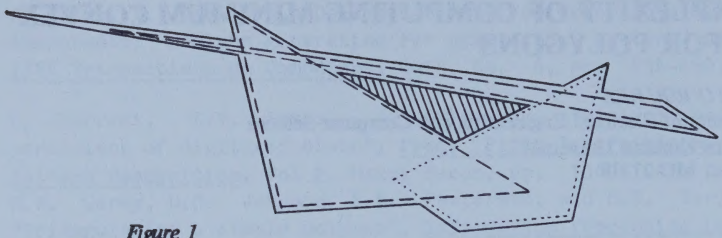


Figure 1

struct all possible convex pieces from the vertices, and then check subsets of these pieces for complete coverage. If the search examines smaller cardinality subsets first, the first cover found will be a minimum cover. Clearly this algorithm requires exponential time.

If Steiner points are allowed, it seems natural to restrict the convex pieces to be bounded by extensions of the polgon's edges (see Figure 1). Pavlidis observed that only *maximal* convex subsets need to be considered [17]: those which are not enclosed within any other internal convex subset. It is then clear that there are only a finite number of possible covers, and a searching algorithm is possible. Pavlidis investigated such algorithms for pattern recognition applications [17-20], although he was not interested in minimum covers. In [11] it is shown that searching for minimum covers under the edge-extension restriction can be accomplished by minimizing a Boolean expression, a known NP-complete problem. This does not, however, establish that the restricted minimum cover problem is NP-complete, and the complexity of this problem remains unknown.

A special case where the restriction to edge extensions is especially appropriate is the class of *rectilinear* polygons: those whose edges are parallel to one of two orthogonal directions. The restriction in this case implies that each piece must be a rectangle. Again it is clear that the problem is decidable by a searching algorithm. Masek showed that finding a minimum rectangle cover of a multiply-connected rectlinear polygon is NP-complete [9]. Recently, Chaiken *et al* have shown that if the class of rectlinear polygons is further restricted to be "convex" in the sense that every horizontal and vertical line meets the region in an interval, then a minimum rectangle cover can be found in polynomial time [1]. The problem for the important class of simply-connected rectilinear polygons remains open.

3. Counterexamples to Steiner Point Restrictions

Although the restriction that the pieces be bounded by edge extensions may seem natural for pattern recognition applications, it is a true restriction, in that there are polygons whose minimum covers require the use of pieces that are not bounded by such edge extensions. In fact there are rectilinear polygons that need edges in their minimum covers that are neither horizontal or vertical. Figure 2 shows a simply-connected example. As 9 mutually invisible points can be located in the interior of the polygon, any convex cover must use at least 9 pieces. The eight horizontal and vertical rectangles shown in Figure 3 can be proven to be necessary, leaving the ninth piece to use diagonal edges, as illustrated [12].

Although the diamond region in Figure 3 is not bounded by extensions of the polygon's edges, its vertices can be chosen to lie at intersections of edge extensions. It is natural to ask, then, whether restricting the Steiner points in this way is a true restriction. In fact it is, as can be verified by merging two copies of the polygon of Figure 2; the result is shown in Figures 4 and 5. The left diamond region requires a vertex somewhere along the segment labeled λ in Figure 5, but no two edge extensions intersect there.

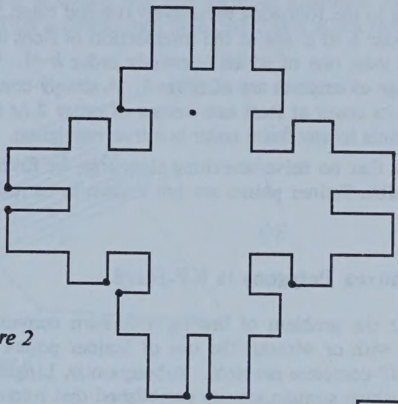


Figure 2

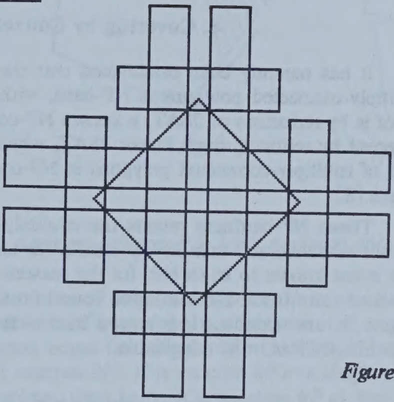


Figure 3

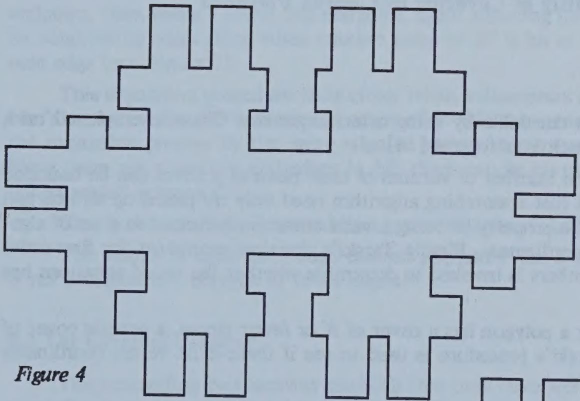


Figure 4

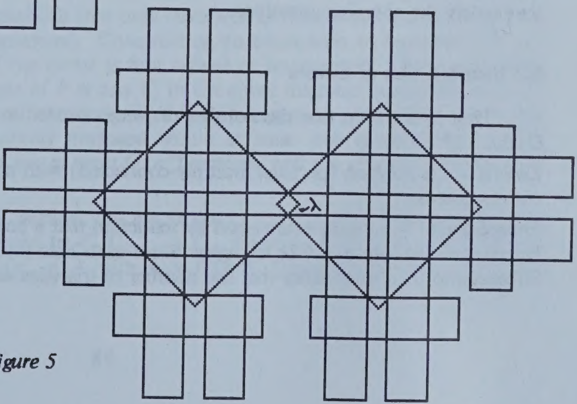


Figure 5

One could also restrict Steiner points to the following recursively defined class: a vertex of the polygon is of order 0; a point is of order k iff it lies at the intersection of lines determined by points of no more than order $k-1$, at least one of which is exactly order $k-1$. Under this definition, points at the intersection of edge extensions are of order 1. A simply-connected polygon can be constructed that requires in its cover at least one vertex of order 2 or more [13], and I conjecture that restricting Steiner points to any finite order is a true restriction.

The implication of these examples is that no naive searching algorithm for finding general minimum covers is known, since the possible Steiner points are not known to be restrictable to a finite class.

4. Covering by Convex Polygons is NP-Hard

It has recently been established that the problem of finding minimum convex covers of multiply-connected polygons is NP-hard, with or without the use of Steiner points [15]. The proof is by reduction of 3SAT, a known NP-complete problem. Subsequently, Lingas simplified by proof by reducing from Planar 3SAT, which simultaneously established that minimum partition of multiply-connected polygons is NP-complete, again with or without the use of Steiner points [8].

These NP-hardness results use multiply-connected polygons in their proofs, and so they do not establish any complexity results for simply-connected polygons. Also, the cover problem is not known to be in NP, for the reasons discussed in the previous two sections. Thus the NP-hard result establishes a lower bound on the complexity; it is possible for an NP-hard problem to be undecidable. It is shown in the next section that, unlike several infinite tiling problems [7], CCP is in fact decidable.

5. Decidability of Covering by Convex Polygons

5.1 Proof Sketch

We now show that CCP is decidable by using a technique that Chazelle employed on a similar problem [2]. A proof sketch is as follows [14].

It is first established that the number of vertices of each piece of a cover can be bounded by a function of n. This ensures that a searching algorithm need only try pieces up to a certain size. Secondly it is shown that the property of being a valid cover is equivalent to a set of algebraic equations in the vertex coordinates. Finally Tarski's decision procedure for first-order sentences in the field of real numbers is invoked to determine whether the set of equations has a solution.

Thus to determine whether a polygon has a cover of K or fewer pieces, a generic cover of K pieces is constructed, and Tarski's procedure is used to see if there exist vertex coordinates that satisfy the covering equations..

5.2 Bounded Size of Covers

First it is shown that the number of pieces needed to cover a polygon is never more than $O(n)$.

Lemma 1. A polygon (perhaps multiply-connected) with n vertices has a cover with $\leqslant 5n/3-4$ convex pieces.

Proof Sketch. It is easily established by induction that a polygon with n vertices and h holes can be triangulated into $n-2+2h$ triangles. Since each hole must have at least 3 vertices, $n \geqslant 3+3h$. Substitution then establishes that the number of triangles satisfies the bound in the Lemma. □

Next it is shown that any piece of a convex cover never needs to have more than $O(n)$ vertices. This is established by expanding a given piece as much as possible.

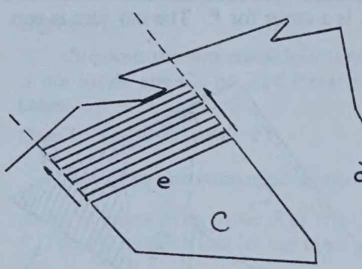


Figure 6

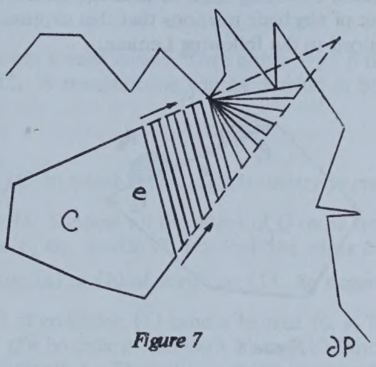


Figure 7

Lemma 2. Any convex polygonal subset C of an n vertex polygon P is itself enclosed within a subset C' that has no more than n vertices.

Proof. Perform the following expansion procedure for each edge e of C. Let ∂P be the boundary of P. If e does not touch ∂P, move it parallel to itself away from the centroid of C, adjusting its length to mate with extensions of its adjacent edges (see Figure 6). Continue this motion until either its length is reduced to zero or it touches ∂P. If it touches ∂P at a single point that is not an endpoint of e or at two or more points, then halt. If it touches ∂P at just one endpoint, then swing e about this endpoint, again adjusting its length to match the extension of its neighboring edge. Stop when another point of ∂P is hit or e becomes collinear with its adjacent edge (see Figure 7).

This expansion procedure halts either when e disappears or it abuts against ∂P. The possible "abutments" are the vertices and edges of P. No two non-collinear edges can be stopped in the expansion process by the same edge or vertex, nor even by an adjacent edge and vertex. Since there are n vertices and edges in ∂P, there can be no more than n non-collinear edges in the expanded polygon C'.

Clearly the method of construction guarantees that $C \subseteq C' \subset P$. □.

This bound is tight: an n edge convex polygon P has a convex subset, namely itself, that is not enclosed in a polygon of fewer edges.

5.3 The Covering Property

The proceeding two lemmas establish that only covers of at most $O(n)$ pieces, each of at most $O(n)$ vertices, need to be considered. Constructing an expression to represent the property of covering will be simpler if the cover is first *refined* or fragmented. Given any cover, introduce vertices wherever two edges of P or any C_i in the cover intersect (and there is not already a vertex). Make the introduced vertices part of both intersecting parts. This fragmentation ensures that every edge is entirely enclosed within at lease one convex piece. After refinement, each of the $O(n)$ pieces has at most $O(n^2)$ vertices, and the polygon P has at most $O(n^2)$ vertices.

Let a refined polygon P have vertices $(a_1,b_1), (a_2,b_2), \cdots, (a_m,b_m)$, with $m = O(n^2)$, and let a refined cover C be represented by $C_1, C_2, \cdots, C_k$, where each C_i is a convex polygon with vertices $(x_1^i,y_1^i), (x_2^i,y_2^i), \cdots, (x_{m_i}^i,y_{m_i}^i)$, with $m_i = O(n^2)$. Boundary lists are assumed to be oriented so that traversal always keeps the interior to the left. Thus outer boundary chains are oriented counterclockwise, and hole chains clockwise. All sets will be considered closed regions, including their boundaries, unless otherwise mentioned. The next task is to construct a set of algebraic relations that that express the fact that **C** is a cover for P. The key idea is contained in the following Lemma.

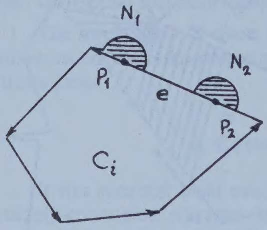


Figure 8

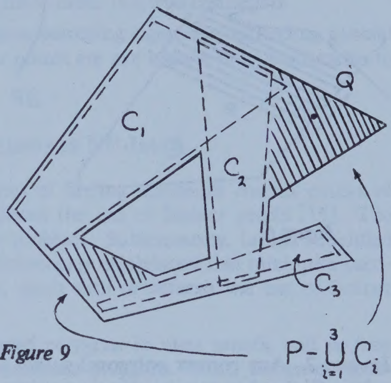


Figure 9

Lemma 3. The following two conditions are necessary and sufficient for **C** to be a refined cover for a refined polygon P:

(1) For every edge e of some $C_i \in \mathbf{C}$, either

 (a) e is equal to some edge of P, oriented in the same direction, or

 (b) e is inside of some C_j (different from C_i), but if it lies on ∂C_j, it is oriented in the opposite direction.

(2) None of the vertices of P are inside the *interiors* of any of the C_i.

Proof.

Necessity. Suppose **C** is a refined cover for P. We will first show that condition (1) holds for an arbitrary edge e of $C_i \in \mathbf{C}$. Let p_1 and p_2 be two arbitrary points on the interior of e, and let N_1 and N_2 be neighborhoods of p_1 and p_2 outside of C_i (see Figure 8). It cannot be the case that N_1 is inside P and N_2 is outside or vice versa, because then ∂P would cross e and the refinement procedure would have split e. Consider each remaining case separately.

Case 1: N_1 *not* $\subseteq P$ and N_2 *not* $\subseteq P$.

Because $p_1 \in C_i$ and $p_2 \in C_i$, and **C** is a cover, $p_1 \in P$ and $p_2 \in P$. Since the neighborhoods are outside of P, p_1 and p_2 lie on ∂P. They must lie on the same edge of P, and e must match this edge exactly, otherwise e would have been refined. The orientations of the edges must match to keep the inside of C_i consistent with the inside of P. Thus clause (a) of condition (1) holds.

Case 2: $N_1 \subseteq P$ and $N_2 \subseteq P$.

Since the points in the neighborhood are in P, and **C** is a valid cover, they must be inside some C_j's. Each neighborhood must be entirely enclosed within one C_j, and in fact the N_1 and N_2 must be enclosed within the same C_j, otherwise e would have been refined. This implies that all of e is enclosed within C_j. Since this C_j encloses neighborhoods outside of C_i, if e is on ∂C_j, then the edge orientations are in opposite directions. Thus clause (b) of condition (1) holds.

Concerning condition (2), if any vertex of P is in the interior of some C_i, then since a neighborhood of this vertex includes points outside of P, C_i encloses points outside of P, contradicting the assumption that **C** is a valid cover.

Sufficiency.

Suppose the two conditions hold true, but **C** is not a valid cover. Then either $\exists p \in P$ that is not inside any C_i, or $\exists p \notin P$ that is inside some C_i. A contradiction will be derived in both cases.

Case 1: $\exists p \in P$ *s.t.* $p \notin C_i\ \forall i$.

Let Q be the connected component of $P - \bigcup_{i=1}^{k} C_i$ in which p lies. Q's boundary is composed of edges from either P or the C_i's (see Figure 9). Suppose all the edges of Q come from P. Then Q must in fact be the same as P, and all the C_i are outside P. But then the edges that form the boundary of $\bigcup_{i=1}^{k} C_i$ do not satisfy either clause (a) or (b) of condition (1). So suppose at least one edge e of Q comes from C_i. Clause (a) of condition (1) cannot be true for e, because the subtraction $P - C_i$ would eliminate e from Q's boundary. Similarly clause (b) cannot be true because the subtraction $P - C_i - C_j$ would eliminate e. Thus all possibilities contradict condition (1).

Case 2: $\exists p \notin P$ *s.t.* $p \in C_i$ for some i.

Let Q be the connected component of $\bigcup_{i=1}^{k} C_i - P$ that includes p. If all the edges of Q come from P, then Q coincides with a hole of P. Any vertex on the boundary of this hole must be inside some C_i, contradicting condition (2). So suppose at least one edge e of Q comes from C_i. This edge cannot satisfy either clause (a) or (b) of condition (1), because in the first case it would be removed by the subtraction of P, and in the second case it would be absorbed by the union with C_j. Thus all cases contradict the lemma. □

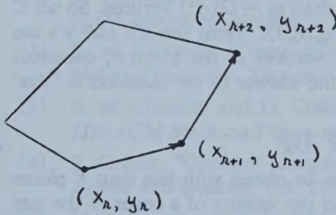


Figure 10

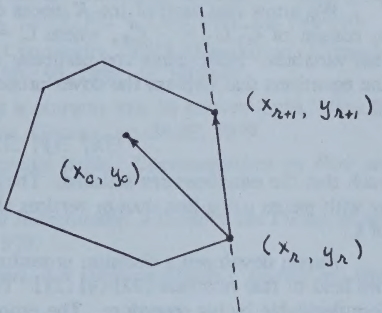


Figure 11

A set of relations for expressing the proposition that **C** is a refined cover for P must then check three conditions:

(1) Each piece C_i must be convex.

(2) Each edge of **C** must satisfy condition (1) of Lemma 3.

(3) Each vertex of P must satisfy condition (2) of Lemma 3.

The first check can be accomplished by computing the cross products of successive edge vectors (see Figure 10). Thus C_i is convex and oriented counterclockwise iff (superscript i's will be dropped when clear from the context)

$$(x_{r+1}-x_r)(y_{r+2}-y_{r+1})-(x_{r+2}-x_{r+1})(y_{r+1}-y_r) \geqslant 0,\ \forall r,\ 1 \leqslant r \leqslant m_i \qquad (1)$$

Clause (a) of condition (1) merely requires a check for equal edges. The edge $e=((x_1,y_1),(x_2,y_2))$ equals an edge of P iff

$$\exists r \text{ s.t. } x_1=a_r\ ,\ y_1=b_r\ ,\ x_2=a_{r+1}\ ,\ y_2=b_{r+1}, \qquad (2)$$

where $((a_r,b_r),(a_{r+1},b_{r+1}))$ is an edge of P.

Clause (b) requires determining whether an edge is inside a convex polygon C_i. It is iff its two endpoints are. A point is inside C_i iff it is inside every half-plane determined by C_i's edges. This half-plane check can be performed by a cross-product. Thus the point (x_0,y_0) is inside C_i (see Figure 11) iff

$$(x_{r+1}-x_r)(y_0-y_r)-(x_0-x_r)(y_{r+1}-y_r) \geqslant 0,\ \forall r,\ 1 \leqslant r \leqslant m_i \qquad (3)$$

The check that no vertex of P should be inside any C_i can be made by slight modification of equation (3).

It should now be clear that the property that **C** is a cover for P can be expressed by a series of algebraic equations of the form (1), (2), or (3), coupled by the logical connectives *and* and *or*.

5.4 Tarski's Decidability Theorem

The bounds derived in Section 3 and the equations constructed in Section 4 can be combined to write a single algebraic relation that precisely captures the question "Is there a cover of P with K or fewer convex pieces?"

We know that each of the K pieces cannot have more than $m=O(n^2)$ vertices. So set **C** to consist of $C_1, C_2, \cdots, C_K$, where $C_i=((x_1^i,y_1^i), \cdots, (x_m^i,y_m^i))$. Here the x's and y's are real variables. Now, using the particular integer coordinate vertices of the given P, construct the equations that express the cover property for **C**. Then the answer to the question is "Yes" iff

$$\exists x_1^1\ \exists y_1^1\ \exists x_2^1\ \exists y_2^1 \cdots \exists x_K^m\ \exists y_K^m$$

such that the equations are satisfied. The fact that there may be covers with less than K pieces or with pieces using less than m vertices clearly will not affect the validity of a cover of the size of **C**.

Tarski developed a decision procedure for all first-order sentences of no free variables in the field of real numbers [22] [4] [21]. The expression above is such a sentence, and is therefore decidable by his procedure. The procedure eliminates quantifiers and reduces the sentence to a single polynomial relation of no variables, which is easily checked for validity. Modern formulations of Tarski's procedure [10] appear to require $O(2^{2^n})$ time. Despite this slowness, the procedure establishes the decidability of the minimum convex cover problem:

Theorem. The minimum convex cover problem is decidable.

6. Discussion

The proof of the preceding section can easily be modified to hold under any of the various restrictions we have discussed, such as to simply-connected polygons or to rectangular pieces. Thus all these minimum covering (and partition) problems are decidable.

The current state of knowledge on the complexity of the various problems is displayed in the Table below.

Polygon Class	Components	Cover	Partition
multiply-connected	convex	NP-hard	NP-complete
simply-connected	convex	?	$O(n^3)$
rectlinear, multiply-connected	rectangular	NP-complete	polynomial
rectilinear, simply-connected	rectangular	?	$O(n^3)$

It is an outstanding open problem to determine whether a simply-connected polygon can be minimally covered in polynomial time, even when restricted to the rectilinear case. It would also be of interest to derive some restrictions on the class of convex pieces that are required in general minimum covers. For example, if the vertices of the original polygon have rational coordinates, are irrational coordinates ever required in a minimum cover?

Acknowledgements

I am indebted to Kenneth J. Supowit and the participants of a McGill University Compuational Geometry seminar for clarifying discussions.

REFERENCES

[1] S. Chaiken, D. J. Kleitman, M. Saks, and J. Shearer, "Covering Regions by Rectangles," *SIAM Journal on Algebraic and Discrete Methods*, Vol. 2, No. 4, Dec. 1981, pp. 394-410.

[2] B. M. Chazelle, "Computational geometry and convexity," PhD Dissertation, Carnegie-Mellon University, Technical Report CMU-CS-80-150, 1980, pp. 140-145.

[3] B. M. Chazelle and D. Dobkin, "Decomposing a polygon into its convex parts," *Proc. of 11th ACM Symp. on Theory of Computing*, Atlanta, Georgia, pp. 38-48, 1979.

[4] P. Cohen, "Decision procedures for real and p-adic fields," *Communications on Pure and Applied Mathematics*, Vol. 22, 1969, pp. 131-151.

[5] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, San Francisco: W. H. Freeman, 1979.

[6] D. S. Johnson, "The NP-Completeness Column: An Ongoing Guide," *Journal of Algorithms*, June 1982.

[7] H. R. Lewis, *Unsolvable Classes of Quantification Formulas*, Reading, Mass.: Addison-Wesley, 1979.

[8] A. Lingas, "The Power of Non-Rectilinear Holes," to appear in the *Proc. of the 9th Colloquium on Automata, Languages, and Programming*, Aarhus, 1982.

[9] W. J. Masek, "Some NP-complete set covering problems," unpublished manuscript, Aug. 1979; quoted in [5, p. 232].

[10] L. Monk, "Elementary-recursive decision procedures," PhD Thesis, University of California, Berkeley, 1975.

[11] J. O'Rourke, "Polygon decomposition and switching function minimization," *Computer Graphics and Image Processing*, Vol. 19, pp. 384-391, Aug. 1982.

[12] J. O'Rourke, "Minimum Convex Covers: Some Counterexamples," Johns Hopkins University, Department of Electrical Engineering and Computer Science, Technical Report JHU-EECS 82-1, Jan. 1982.

[13] J. O'Rourke, "A Note on Minimum Convex Covers for Polygons," Johns Hopkins University, Department of Electrical Engineering and Computer Science, Technical Report JHU-EECS 82-3, Mar. 1982.

[14] J. O'Rourke, "The Decidability of Covering by Convex Polygons," Johns Hopkins University, Department of Electrical Engineering and Computer Science, Technical Report JHU-EECS 82-4, May 1982.

[15] J. O'Rourke and K. J. Supowit, "Some NP-hard polygon decomposition problems," to appear in *IEEE Trans. on Information Theory*, Mar. 1983.

[16] L. Pagli, E. Lodi, F. Luccio, C. Mugnai, and W. Lipski, "On two dimensional data organization 2," *Fundamenta Informaticae*, vol. 2, pp. 211-226, 1979.

[17] T. Pavlidis, "Analysis of set patterns," *Pattern Recognition*, vol. 1, pp. 165-178, 1968.

[18] T. Pavlidis, "Representation of figures by labelled graphs," *Pattern Recognition*, vol. 4, pp. 5-17, 1972.

[19] T. Pavlidis, "Structural pattern recognition: Primitives and juxtaposition relations," in *Frontiers of Pattern Recognition* (Ed. S. Watanabe), New York: Academic Press, pp. 421-451, 1972.

[20] T. Pavlidis, *Structural Pattern Recognition*, pp. 236-241, Berlin-Heidleberg-New York: Springer Verlag, 1977.

[21] M. O. Rabin, "Decidable theories," in *The Handbook of Mathematical Logic*, ed. J. Barwise, North-Holland, 1978, pp. 595-629.

[22] A. Tarski, "A decision method for elementary algebra and geometry," second edition, revised, University of California press, 1951.

[23] G. T. Toussaint, "Pattern recognition and geometrical complexity," *Proc. of 5th International Conference on Pattern Recognition*, Miami Beach, Florida, pp. 1324-1347, Dec. 1980.

AN OPTIMAL ALGORITHM TO CONSTRUCT ALL VORONOI DIAGRAMS FOR K NEAREST NEIGHBOR SEARCHING IN THE EUCLIDEAN PLANE

Frank Dehne
Lehrstuhl für Informatik I
Bayerische Julius-Maximilians-Universität Würzburg
D-8700 Würzburg

ABSTRACT

This paper presents an algorithm, that constructs all Voronoi diagrams for k nearest neighbor searching simultaneously. Its space and time complexity of $O(N^4)$ is shown to be optimal.

1. INTRODUCTION

In [6] Shamos and Hoey introduce the idea of generalized Voronoi diagrams to get an optimal solution of the k nearest neighbor problem and give an O(N logN) algorithm to construct the order one diagram.
Lee ([2]) extends this to an algorithm, that computes an order k diagram in $O(k^2 N \log N)$.
To answer k nearest neighbor queries with arbitrary k we now want to construct all Voronoi diagrams.
This paper presents a simple solution of this problem.
The given algorithm has time and space complexity $O(N^4)$ and is shown to be optimal. Its implementation is not very difficult and the constant factors for the complexity are expected to be quite good.

2. K NEAREST NEIGHBOR SEARCHING AND GENERALIZED VORONOI DIAGRAMS

Let $S:=\{s_1,\dots,s_N\}$ be a set of $N\geqq 3$ points in the Euclidean plane $\mathbb{E}^2$ (with distance measure d). We shall assume that no more than three of these points lie on a circle and that they are not all collinear.
To answer a query for k nearest neighbors of a point $q\in\mathbb{E}^2$, we have to find a subset $A\subset S$ with $|A|=k$ and $(\forall x\in A, y\in S-\{a\}):d(q,x)\leqq d(q,y)$.
With $B(x,y):=\{z\in\mathbb{E}^2/d(x,z)=d(y,z)\}$ and $h(x,y):=\{z\in\mathbb{E}^2/d(x,z)\leqq d(y,z)\}$ we call $v(A):=\bigcap_{\substack{x\in A\\ y\in S-A}} h(x,y)$ the Voronoi polygon of $A\subset S$ and $V_k(S):=\{v(A)/A\subset S \wedge |A|=k\}-\{\phi\}$ the (generalized) Voronoi diagram of order k.
It is easy to see that $V_k(S)$ can be described by a straight line graph, that divides the Euclidean plane into a finite number of convex polygons* (the Voronoi polygons) and that $A\subset S$ is a set of k nearest neighbors of all query points $q\in v(A)\in V_k(S)$.
With these postulates we can solve the k nearest neighbor problem in the following way:

(A) Construct all $V_k(S)$ for $1\leqq k\leqq N-1$ (preprocessing).

(B) For every query $(q,k)\in\mathbb{E}^2\chi\{1,\dots,N-1\}$ find a $v(A)\in V_k(S)$ with $q\in v(A)$.

For part (B) Kirkpatrick ([1]) has already found an optimal algorithm that answers a query in O(k+logN) steps.
The next two sections of this paper will describe an optimal solution of part (A).

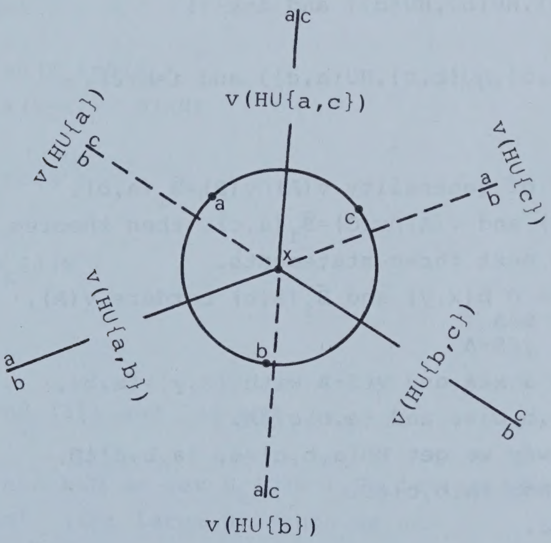
---

* In the following this paper only will operate with this diagram also called $V_k(S)$.

3. PROPERTIES OF GENERALIZED VORONOI DIAGRAMS

Every Voronoi edge (edge of a Voronoi diagram) is part of a bisector $B(x,y)$ with $x,y \in S$. So let $\bar{B}_k(x,y)$ be the part of $B(x,y)$ that is Voronoi edge of $V_k(S)$.
With every Voronoi polygon being convex and only up to three points of S lying on a circle, every Voronoi point (point of a Voronoi diagram) has degree three and is the center of exactly three points of S.
Now we can prove the following theorem.

Theorem 1:
Let $x \in \mathbb{E}^2$ be the center of $a,b,c \in S$ and $H := \{z \in S / d(x,z) < d(x,a)\}$ with $|H| =: k \leqq N-3$, then x is Voronoi point of $V_{k+1}(S)$ and $V_{k+2}(S)$.
The Voronoi edges and polygons that are incident upon x are given by the following diagram.



$\frac{a}{b}$ ——— : $\bar{B}_{k+1}(a,b)$

$\frac{a}{b}$ – – – : $\bar{B}_{k+2}(a,b)$

Proof:

$H\cup\{a\}$, $H\cup\{b\}$ and $H\cup\{c\}$ are sets of $k+1$ nearest neighbors of x. So $x\in v(H\cup\{a\})\cap v(H\cup\{b\})\cap v(H\cup\{c\})$ is a Voronoi point of $V_{k+1}(S)$. Because $H\cup\{a,b\}$, $H\cup\{b,c\}$ and $H\cup\{a,c\}$ are sets of $k+2$ nearest neighbors of x, it is also a Voronoi point of $V_{k+2}(S)$. With this the construction of the above diagram is trivial.

$\square$

The next theorem will demonstrate, that every Voronoi point can be constructed as described in theorem 1.

Theorem 2:

Let a Voronoi point $x\in v(A)\cap v(B)\cap v(C)$ with $v(A),v(B),v(C)\in V_i(S)$ be the center of $a,b,c\in S$ and $H:=\{z\in S/d(x,z)<d(x,a)\}$ with $|H|:=k\leq N-3$, then

$(\{A,B,C\}=\{H\cup\{a\},H\cup\{b\},H\cup\{c\}\}$ and $i=k+1)$

or

$(\{A,B,C\}=\{H\cup\{a,b\},H\cup\{b,c\},H\cup\{a,c\}\}$ and $i=k+2)$.

Proof:

Let without loss of generality $v(A)\cap v(B)=\overline{B}_i(a,b)$, $v(B)\cap v(C)=\overline{B}_i(b,c)$ and $v(A)\cap v(C)=\overline{B}_i(a,c)$, then theorem 2 follows from the next three statements.

(1) Because $v(A)=\bigcap\limits_{\substack{x\in A\\ y\in S-A}} h(x,y)$ and $\overline{B}_i(a,b)$ borders $v(A)$,

there exists a $x\in A$ and $y\in S-A$ with $\{x,y\}=\{a,b\}$, getting $A\cap\{a,b,c\}\neq\phi$ and $\{a,b,c\}\not\subset A$.

In the same way we get $B\cap\{a,b,c\}\neq\phi$, $\{a,b,c\}\not\subset B$, $C\cap\{a,b,c\}\neq\phi$ and $\{a,b,c\}\not\subset C$.

(2) $H\subset A$, $H\subset B$, $H\subset C$.

Proof of $H\subset A$ with $a\in A$ and $b\notin A$ (see (1)):

If there would be a $z\in H$ with $z\notin A$, this would be a contradiction to $x\in v(A)$ because of $d(x,z)<d(x,a)$.

(3) $A,B,C\subset H\cup\{a,b,c\}$.

Proof of $A\subset H\cup\{a,b,c\}$ with $a\in A$ and $b\notin A$ (see (1)):

if there would be a $z \in A$ with $z \notin H \cup \{a,b,c\}$, we would have $d(x,z) > d(x,a) = d(x,b)$ with $z \in A$ and $b \notin A$;
a contradiction to $x \in v(A)$.

$\lvert\overline{\underline{\ \ }}\rvert$

With theorem 2 we know theorem 1 describing all Voronoi points, edges and polygons of all $V_k(S)$ $(1 \leq k \leq N-1)$. This is the main idea for the algorithm in section 4. For the analysis of this algorithm the next theorem gives us the number of Voronoi points, edges and polygons we have to compute.

Theorem 3:

Let I_k be the number of Voronoi points, E_k the number of edges and N_k the number of polygons of $V_k(S)$ $(1 \leq k \leq N-1)$, then we get

$$\theta(I_k) = \theta(E_k) = \theta(N_k) \quad ^* \qquad (1)$$

$$N_k \in O(k(N-k)) \subset O(kN) \qquad (2)$$

$$\sum_{k=1}^{N-1} N_k \in \theta(N^3) \qquad (3)$$

$$\sum_{k=1}^{N-1} kN_k \in \theta(N^4) \qquad (4).$$

Proof:

(1), (2) and (3): see [2] and [6].

(4):

From (2) and $k \leq N$ we get $N_k \in O(N^2)$. So let us take an $a \in \mathbb{N}$ with $N_k \leq aN^2$ (for large N), then we get

---

* Having two functions $f,g: \mathbb{N} \to \mathbb{R}$ we say
$f \in O(g) :\Longleftrightarrow (\exists m \in \mathbb{N}, c \in \mathbb{N})(\forall n \geq m): f(n) \leq c \cdot g(n)$,
$f \in \Omega(g) :\Longleftrightarrow (\exists m \in \mathbb{N}, c \in \mathbb{N})(\forall n \geq m): f(n) \geq c \cdot g(n)$ and
$f \in \theta(g) :\Longleftrightarrow (f \in O(g) \wedge f \in \Omega(g))$.

$$\sum_{k=1}^{N-1} kN_k \leqq \sum_{k=1}^{N-1} kaN^2 = \frac{a}{2}(N-1)N^3 \in O(N^4).$$

To prove $\sum_{k=1}^{N-1} kN_k \in \Omega(N^4)$ we define $I := \{1 \leqq k \leqq N-1 / N_k \in \Omega(N^2)\}$ and $J := \{1 \leqq k \leqq N-1 / N_k \notin \Omega(N^2)\}$ and show $|I| \in \Omega(N)$.

Assuming $|I| \notin \Omega(N)$ we would get $\sum_{k \in I} N_k \notin \Omega(N^3)$ from $N_k \in O(N^2)$.

By definition of J we get $\sum_{k \in J} N_k \notin \Omega(N^3)$ and further more

$$\sum_{k=1}^{N-1} kN_k = \sum_{k \in I} N_k + \sum_{k \in J} N_k \notin \Omega(N^3);$$ a contradiction to (3).

Let b and c be two numbers with $|I| \geqq bN$ and $N_k \geqq cN^2$ for $k \in I$ and large N, then we get

$$\sum_{k=1}^{N-1} kN_k \geqq \sum_{k \in I} kN_k \geqq cN^2 \sum_{k \in I} k \geqq cN^2 \sum_{k=1}^{I} k \geqq cN^2 \sum_{k=1}^{bN} k$$

$$= \frac{cb}{2} N^3 (bN+1) \in \Omega(N^4).$$

$\Box$

4. THE ALGORITHM

This section will give a description of the algorithm to construct all $V_k(S)$.

It needs $\theta(N^4)$ time and $\theta(N^4)$ storage. Equation (4) of theorem 3 showed that all Voronoi diagrams have a space complexity of at least $\theta(N^4)$ and therefore we need at least $\theta(N^4)$ steps to construct them. So the algorithm has optimal time and space complexity.

The data structure for the $V_k(S)$ is the same Kirkpatrick defines for the input data of his region location algorithm (see [1]). Every $V_k(S)$ is stored by a list of its Voronoi points, each of wich contains the information about the incident Voronoi edges and polygons.

The basic idea is, to take all triples of points $a,b,c \in S$ and compute all Voronoi points in the way theorem 1 describes. Theorem 2 makes sure, that we get all of them. In a second step we have to link the points of each $V_k(S)$ together.

Before coming to a detailed description of the algorithm, we need some more definitions.

Definition:

For every $(s_a, s_b, s_c) \in S^3$ let

(1) $M(a,b,c)$ be the center of s_a, s_b, s_c (if exists)

(2) $H(a,b,c) := \{y \in S - \{s_a, s_b, s_c\} / d(M(a,b,c), y) < d(M(a,b,c), s_a)\}$

(3) $\nabla := \{(u,v,w) \in \{1,\ldots,N\}^3 / u > v > w\}$.

With this we can construct all $V_k(S)$ as follows:

(1) Construct an array L of all $M(a,b,c)$ with $(a,b,c) \in \nabla$ in wich every $M(a,b,c)$ can be found in $O(1)$ steps (store $M(a,b,c)$ with $(a,b,c) \in \nabla$ at the adress $\binom{a-1}{3} + \binom{b-1}{2} + c$).

(2) Traverse L. For every $M(a,b,c)$ calculate $H(a,b,c)$ and the incident rays and polygons as described by theorem 1 and add them to the lists of $V_{|H(a,b,c)|+1}(S)$ and $V_{|H(a,b,c)|+2}(S)$. Note the two adresses in L.

(3) Traverse L again. Every $M(a,b,c)$ is a Voronoi point in two lists $V_i(S)$ and $V_{i+1}(S)$ with at most 6 incident rays. With each ray r do the following steps: Let r be in $V_j(S)$ and be part of $B(s_a, s_b)$. Take all $M(a,b,x)$ with $s_x \in S - \{s_a, s_b, s_c\}$ and check, whether $M(a,b,x)$ is a Voronoi point of $V_j(S)$ and lies on r. If there are more such points take the $M(a,b,x_0)$ with minimum distance from $M(a,b,c)$. Reduce r to an edge $(M(a,b,c), M(a,b,x_0))$ and the corresponding ray of $M(a,b,x_0)$ to an edge $(M(a,b,x_0), M(a,b,c))$.

With theorem 3 the analysis of the algorithm is easy. The space complexity is

$$\theta\left(\sum_{k=1}^{N-1} (I_k + kE_k)\right) = \theta\left(\sum_{k=1}^{N-1} kN_k\right) = \theta(N^4).$$

L contains $\theta(N^3)$ points. So part (1) of the algorithm needs time $\theta(N^3)$. In part (2) for each of these points we need $\theta(N)$ steps and so the whole part takes $\theta(N^4)$ steps. In the same way you see evidently part (3) needing time $\theta(N^4)$ too. So the time complexity of the whole algorithm is $\theta(N^4)$.

This short description of the algorithm yields already so many details, that it is easy to be implemented. We don´t need very much overhead (compared with the algorithms in [2] and [6]) and the multiplicative constants for the complexity are expected to be quite good.

5. ACKNOWLEDGMENT

The author wishes to thank Prof. Dr. H. Noltemeier for his encouragement and helpfull discussions.

6. REFERENCES

[1] Kirkpatrick: OPTIMAL SEARCH IN PLANAR SUBDIVISIONS, Department of Computer Science, Univ. of British Columbia, Vancouver, B.C., Canada, Report 81-13, 1981

[2] Lee: AN APPROACH TO FINDING THE K-NEAREST NEIGHBOR IN THE EUCLIDEAN PLANE, Department of Electrical Engeneering and Computer Science, Northwestern Univ., Evanston, Il. 60201, USA, Report 1981

[3] Maurer: THE POST OFFICE PROBLEM AND RELATED QUESTIONS, in Noltemeier (ed.): GRAPHTHEORETIC CONCEPTS IN COMPUTER SCIENCE, Proceedings of the International Workshop WG80, Bad Honnef, June 1980, Lecture Notes in Computer Science 100, Springer 1981

[4] Shamos: COMPUTATIONAL GEOMETRY, Yale University, New Haven, Connecticut, USA, Ph.D.Thesis, 1975

[5] Shamos: GEOMETRIC COMPLEXITY, Proceedings of the 7th Annual ACM Symposium on Theory of Computing, 1975, pp.224-233

[6] Shamos,Hoey: CLOSEST-POINT PROBLEMS, Proceedings of the 16th IEEE Symposium on Foundations of Computer Science, 1975, pp. 151-162

THE COMPLEXITY OF VORONOI DIAGRAMS IN HIGHER DIMENSIONS

RAIMUND SEIDEL
Computer Science Department
Cornell University

ABSTRACT

An exact bound is given for the number of faces in closest point Voronoi diagrams of n points in Euclidean d-space E^d. We also give an asymptotically tight bound for the number of faces in furthest point Voronoi diagrams in E^d.

INTRODUCTION

Let T be a finite subset of E^d. For pϵT the set $V_p(T):=\{x \epsilon E^d | d(x,p) \leq d(x,q)$ for all $q \epsilon T\}$ is called the **closest point V-region** of p w.r.t. T and the set $W_p(T):=\{x \epsilon E^d | d(x,p) \geq d(x,q)$ for all $q \epsilon T\}$ is called the **furthest point V-region** of p. The **closest point** (resp. **furthest point**) **Voronoi diagram** of T, $VOD_c(T)$ (resp. $VOD_f(T)$), is the collection $\{V_p(T) | p \epsilon T\}$ (resp. $\{W_p(T) | p \epsilon T\}$). Closest and furthest point V-regions are convex polyhedra and the union of the V-regions in $VOD_c(T)$ (resp. $VOD_f(T)$) covers E^d.

For $0 \leq k \leq d$, let $\Phi_k(T)$ (resp. $\Psi_k(T)$) denote the number of sets S such that S is a k-face of one of the polyhedra of $VOD_c(T)$ (resp. $VOD_f(T)$). Let $M_k(n,d)$ and $N_k(n,d)$ be the maximum of $\Phi_k(T)$ and $\Psi_k(T)$ respectively. We will prove that $M_k(n,d) = f_{d-k}(n,d+1) - \delta_{0k}$, where $f_i(n,d)$ denotes the maximal number of i-faces for d-dimensional convex polytopes with n vertices. Furthermore we will show that $N_k(n,d) \leq f_{d-k}(n,d+1)$ and that for fixed d>1 $N_0(n,d) = \Theta(n^{\lceil n/2 \rceil})$. It is known that for $0 \leq i < d$

$$f_i(n,d) = \frac{n-(d-2s)(n-i-2)}{n-i-1} \sum_{j=0}^{s} \binom{n-j-1}{i-j+1} \binom{n-i-1}{2j-i-1+d-2s},$$

where $s = \lfloor d/2 \rfloor$, and that these numbers are realized by all simplicial neighborly polytopes (see [3, 9.6] and [5]). Also note that for fixed d, $f_{d-1} = \Theta(n^{\lfloor d/2 \rfloor})$. It is assumed that the reader is familiar with such terms as polytope, k-face, simplicial, neighborly, etc. For definitions of these terms see [3].

VORONOI DIAGRAMS AND POLYTOPES

In this section we describe a transform due to K.Q.Brown [1], which relates Voronoi diagrams in E^d and (d+1)-polytopes with the property that all their vertices lie on one sphere. Let $T \subset E^d$ be finite. Embed E^d in a hyperplane H of E^{d+1} which does not contain the origin O. Consider the stereographic projection $g: E^{d+1} - \{O\} \rightarrow E^{d+1} - \{O\}$: $g(x) = \frac{x}{<x,x>}$, where $<x,y>$ denotes the ordinary scalar product. Note that $g(g(x)) = x$ and that g maps H onto a sphere $S \subset E^{d+1}$ which contains O. Let $T' = \{g(p) | p \epsilon T\}$ and let P be the polytope formed by the convex hull of T'. Note that $O \notin$ P. For a point x $\notin$ P we call a facet T of P a **σ-facet** w.r. to x iff the closed halfspace determined by the affine hull of F, which contains P but does not contain x. We call F a **ρ-facet** w.r. to x iff it is not a σ-facet and the affine hull of F does not contain x. The following theorem is of great importance for our purposes. For a proof see [1] and [7].

Theorem 1: Let T and P be as definded above and let $0 \leq k < d$.
There is a 1-1 correspondence between the set of k-faces in $VOD_c(T)$ (resp. $VOD_f(T)$) and the set of (d-k)-faces of P which are contained in a facet F of P such that F is a ρ-facet (σ-facet) w.r. to O.

THE BOUND FOR CLOSEST POINT VORONOI DIAGRAMS

Theorem 2: For n>d>0 and $0 \leq k < d$ $M_k(n,d) = f_{d-k}(n,d+1) - \delta_{0k}$.

Proof: From Theorem 1 it follows immediately that $M_k(n,d) \leq f_{d-k}(n,d+1)$. For the special case k=0 note that P must have at least one σ-facet w.r. to O as $O \notin$ P.
Now it remains to show that for every n>d>0 there is an n-membered set $T \subset E^d$ such that $M_k(T) = f_{d-k}(n,d+1) - \delta_{0k}$. The strategy to find such a set will be as follows: Consider a sphere S

in E^{d+1}. Judiciously choose an n-membered set T'⊂S and a point q∈S such that P, the convex hull of T', is a simplicial neighborly polytope and has exactly one σ-facet w.r. to q. Thus for k<d, each k-face of P is contained in at least one ρ-facet of P. Now translate S such that q becomes O and apply the stereographic projection g. T' is thus mapped to a set T contained in a hyperplane of E^{d+1} which can be identified with E^d. The d-dimensional closest point Voronoi diagram of T then has $f_{d-k}(n,d+1)-\delta_{0k}$ k-faces by the properties of P and because of Theorem 1. It only remains to show how such a (d+1)-polytope can be constructed.

For even D=2s consider the curve $C_D = \{m_D(t)=(cos t, sin t, cos 2t, sin 2t, ..., cos\ st, sin\ st) \mid 0 \leq t < 2\pi\}$ and for odd D=2s+1 the curve $C_D = \{m_D(t)=g((1,t,t^2,...,t^{2s})) \mid 0 \leq t < \infty\}$. For any D>1, C_D lies on a sphere S in E^D and the convex hull of any n-membered subset of C_D, n>D, is a simplicial neighborly polytope ([3, p.67],[7]).

Choose distinct points $p_1,...,p_D$ from C_D and let F be their convex hull. Choose a point w from the relative interior of F and a point $p_{D+1} \in C_D - F$, where $p_{D+1}=m_D(t')$ for some t'. Let q be the intersection point of S and the straight line passing through p_{D+1} and w which is not equal to p_{D+1}. Clearly there is some $\epsilon>0$ s.t. for all t with $|t-t'|<\epsilon$ the straight line segment connecting $m_D(t)$ and q intersects F in its relative interior. Choose n-D-1 distinct values t_i, with $t_i \neq t'$ and $|t_i - t'| < \epsilon$, and let $p_i = m_D(t_i)$, D+1<i≤n. Let $T'=\{p_1,...,p_n\}$, and let P be the convex hull of T'. Then F is the only σ-facet of P w.r. to q and all other facets of P are ρ-facets. This completes the proof.

THE BOUND FOR FURTHEST POINT VORONOI DIAGRAMS

Theorem 3:

(1) $N_k(n,d) \leq f_{d-k}(n,d+1)$ for n>d>1 and 0≤k<d.
(2) $N_0(n,d) = \Theta(n^{\lceil d/2 \rceil})$ for fixed d>1.

Proof (sketch): (1) follows immediately from Theorem 1. Hence, as $f_d(n,d+1)=\Theta(n^{\lceil d/2 \rceil})$ for fixed d, $N_0(n,d)=O(n^{\lceil d/2 \rceil})$.

Now it remains to show that $N_0(n,d)=\Omega(n^{\lceil d/2 \rceil})$. It is known that $N_0(n,2)=n-1=\Omega(n)$ [8]. Let us consider the case d=3. Let a be a sufficiently large positive real number. Let C' (resp. C") be a unit radius circle in the x_2-x_3- (resp. x_1-x_2-) plane centered at (0,a,0) (resp. (0,-a,0)), and let S' and S" be the semicircles comprising the points of C' and C" respectively whose x_2-coordinates have absolute values greater than a. If one selects sets T' and T" of $\frac{n}{2}$ points on S' and S" respectively, then VOD_f(T'∪T") has at least $(\frac{n}{2}-1)^2$ 0-faces (i.e. vertices). Hence $N_0(n,3)=\Omega(n^2)$.

This construction can be generalized to dimensions d>3 by letting C' and S' be the corresponding respective (d-1)-dimensional unit sphere and hemisphere in the x_2-,...,x_d-hyperplane. For details see [7].

REFERENCES

[1] Brown, K.Q. Voronoi Diagrams from Convex Hulls. Information Processing Letters 9; pp.223-228; 1979.

[2] Dewdney, A.K. and Vranch, J.K. A Convex Partition of R^3 with Applications to Crum's Problem and Knuth's Post-Office Problem. Utilitas Mathematica 12; pp.193-199; 1979.

[3] Gruenbaum, B. *Convex Polytopes.* Wiley Interscience; 1967.

[4] Klee, V. On the Complexity of d-Dimensional Voronoi Diagrams. Archiv der Mathematik 34; pp.75-80; 1980.

[5] McMullen, P. The Maximum Number of Faces of a Convex Polytope. Mathematika 17; pp.179-184; 1970.

[6] Preparata, F. Steps into Computational Geometry. Univ. of Illinois; Urbana, Ill.; Coordinated Science Laboratory; Report R-760; 1977.

[7] Seidel, R. Voronoi Diagrams in Higher Dimensions. Diplomarbeit; Tech. Univ. of Graz, Austria; 1982.

[8] Shamos, M.I. Computational Geometry. Yale Univ.; New Haven, Connecticut; ph.D. Thesis; 1978.

Note: [2], [4], and [6] contain earlier results about the complexity of Voronoi diagrams. They are not referred to in the text.

In E^{d+1}, judiciously choose an n-dimensional set $T' \subset S$ and a point p_0 such that P, the convex hull of T', is a simplicial neighborly polytope and has exactly one n-facet [illegible] to q. [illegible] face of P is contained in at least one n-facet of P. Now translate S such that q becomes [illegible] the stereographic projection g. T' is thus mapped to a set T contained in a hyperplane of E^{d+1} which can be identified with E^d. The n-dimensional closest point Voronoi diagram of T then has $f_k(n,d+1)-\varphi_k$ k-faces by the properties of g and because of Theorem 1. It only remains to show how such a $(d+1)$-polytope can be constructed.

[illegible] D[illegible] C_d lies on a sphere S in E^d and the convex hull of any n-membered subset of C_d [illegible] is a simplicial neighborly polytope ([3], p.67)[7].

Choose distinct points $p_1,\ldots,p_n$ from C_d and let P be their convex hull. Choose a point x from the relative interior of P and a point $p_{n+1} \in C_{d+1}$, where $p_{n+1}=m_d(t')$ for some t'. Let q be the intersectional point of [illegible]

[illegible]

THE [illegible] FURTHEST POINT VORONOI DIAGRAMS

Theorem 2:

(1) $N_k(n,d) \le f_k(n,d+1)$ for $n > d+1$ and $0 \le k < d$

(2) $N_d(n,d) = \Theta(n^{\lceil d/2 \rceil})$ for fixed d.

Proof (sketch): (1) follows immediately from Theorem 1. Hence, as $f_d(n,d+1)=\Theta(n^{\lceil d/2 \rceil})$ for fixed d, $N_d(n,d)=O(n^{\lceil d/2 \rceil})$.

Now it remains to show that $N_d(n,d)=\Omega(n^{\lceil d/2 \rceil})$. [illegible] Let us consider the case $d=3$. Let n be a sufficiently large positive even number. Let C (resp. C') be a [illegible] [illegible] values greater than [illegible]

$VOD_f(C \cup C')$ has at least $(\frac{n}{2})^2$ [illegible]

This construction can be generalized to dimensions $d>3$ by letting C and C' be [illegible] respective [illegible] and hemisphere in the [illegible]

REFERENCES

[1] Brown, K.Q. Voronoi Diagrams from Convex Hulls. Information Processing Letters 9, pp.223-228, 1979.

[2] Dewdney, A.K. and Vranch, J.K. A Convex Partition of R^3 with Applications to Crum's Problem and Knuth's Post-Office Problem. Utilitas Mathematica 12, pp.193-199, 1977.

[3] Grünbaum, B. Convex Polytopes. Wiley Interscience, 1967.

[4] Klee, V. On the Complexity of d-dimensional Voronoi Diagrams. Archiv der Mathematik 34, pp.75-80, 1980.

[5] McMullen, P. The Maximum Number of Faces of a Convex Polytope. Mathematika 17, pp.179-184, 1970.

[6] Preparata, F. Steps into Computational Geometry. Univ. of Illinois, Urbana, Ill. Coordinated Science Laboratory, Report R-760, 1977.

[7] Seidel, R. Voronoi Diagrams in Higher Dimensions. Diplomarbeit, Techn. Univ. of Graz, 1982.

[8] Shamos, M.I. Computational Geometry. Ph.D. Thesis, Yale Univ., 1978.

Note: [2], [4], and [6] contain earlier results about the complexity of Voronoi diagrams. They are not referred to in the text.

A NOTE ON PRECISION AND ILL-POSEDNESS IN LINEAR IMAGING PROBLEMS

Lawrence S. Joyce
Environmental Research
Institute of Michigan
Ann Arbor, MI 48104

William L. Root
Aerospace Engineering Dept.
The University of Michigan
Ann Arbor, MI 48109

ABSTRACT

In many instances the determination of an object function from observations (imaging) is an ill-posed linear inversion problem in Hilbert space. Any regularization necessarily implies a loss of precision in the imaging. The precision attainable in a class of such problems is treated here abstractly in terms of a concept called a linear precision guage, which involves essentially an ordered family of linear estimators. An application to a Fourier imaging problem motivated by synthetic aperture radar is given.

INTRODUCTION AND PROBLEM BACKGROUND

The subject of this paper is ill-posed linear inversion or estimation problems in Hilbert space of a type that occur fairly commonly when an object function is to be reconstructed from observed data that result from a linear transformation of the object function. Such a problem will be ill-posed because the inverse mapping that provides the reconstruction is discontinuous. Because of this ill-posedness there must be imperfections of one kind or another in the reconstruction. On the one hand, if it is attempted to determine the object function too precisely the solution is likely to be wildly unstable; on the other hand, an approximate solution that is very stable may not provide the precision that is possible or desired. For optical imaging systems, for example, diffraction-limited performance defines the resolution when traditional reconstruction methods are used. We consider the question of what precision can be attained while maintaining reasonable stability in terms of a concept that is introduced below, called a linear precision guage. The methodology of the linear precision guage is applied to a Fourier transform imaging problem motivated by synthetic aperture radar.

The term, ill-posed problem, is used in the sense of Hadamard. A problem is well-posed if: (1) the problem has a solution; (2) the solution is unique; (3) the solution varies continuously with the data. A problem is ill-posed if it is not well-posed. A linear inverse problem in Hilbert space given by $s = B\sigma$, where the unknown σ belongs to a Hilbert space H_σ, $s \in H_s$ (another Hilbert space) and $B: H_\sigma \to H_s$ is a nondegenerate compact operator, is necessarily ill-posed because B does not have closed range and hence (3) is not satisfied. Of course, condition (2) may be violated also, but it can always be satisfied by fiat by replacing σ with its orthogonal projection on $\mathcal{N}(B)^{\perp}$ ($\mathcal{N}(B)$ = null space of B). If we replace the equation $s = B\sigma$ with the more realistic $z = B\sigma + w$, which is what we actually consider, where w is an error term or noise, then (1) may be violated. This leads to replacing the original inversion with an estimation, say least-squares, or linear unbiased minimum variance (LUMV) if second-moment statistical information is introduced for w. These estimates are satisfactory in finite-dimensional spaces but not in Hilbert space in this case (see [1]), precisely because condition (3) is violated. The estimation problem is ill-posed. This means that one has to settle for a degraded estimate $\hat{\sigma}$ that is less than satisfactory by the usual standards: the true σ must be restricted in some way; or the estimate will be such that it does not converge to the true value as $w \to 0$; or some combination of these will have to be accepted.

A motivating example is given by synthetic aperture radar (SAR) where often at least, with the proper demodulation method, the received data s can be arranged in a format so as to constitute a space-truncated Fourier transform of the complex reflectivity function σ of the object field of finite extent ([2], [3]). In many SAR applications, the pulse repetition frequency of the radar is significantly higher than the Nyquist bandwidth of the scene to be imaged. In this situation, so-called super-resolution methods, whether explicit or iterative, are obviously attractive but may not be trustworthy.

In general any problem in which the noise-free signal can be represented in the form

$$s(\xi) = \int_E b(\xi,x)\sigma(x)dx, \qquad \xi \in F \tag{1}$$

where E and F are bounded sets in n-dimensional space (usually n=1,2 or 3), $\sigma \in L_2(E)$, and b is square-integrable on $E \times F$ and not degenerate, is of the type to be considered. In fact, the integral operator is then Hilbert-Schmidt, hence compact. It is this problem model one would expect to encounter; however a slightly more general model is defined below.

A CONCEPT OF ATTAINABLE PRECISION

The Problem Model

We wish to ascertain what precision can be attained in an estimate of σ in

$$z = s + w = B\sigma + w \tag{2}$$

where s, B, σ are as before, with H_σ and H_s separable, and w is white noise in H_s. The question cannot be answered, of course, until a formal notion of precision is introduced, and then an answer must have a fuzzy interpretation. For the case H_s is a real Hilbert space, the condition that w be white noise in H_s is to mean that it is a uniformly square-integrable weak random variable in H_s with mean zero and covariance operator cI, $c > 0$, I the identity operator (see [4]). That is, w is a linear mapping from H_s^* into the linear space of real random variables on some probability space $(\Omega, \mathcal{A}, P)$ with the properties: (i) $E[w(f)] = 0$, (ii) $\langle f, cg\rangle = E[w(f)w(g)]$, for all $f, g \in H_s^* = H_s$. We can *formally* write condition (iii) as $\langle f, cg\rangle = E\langle w, f\rangle\langle w, g\rangle$, which is perhaps more intuitive, but w is not properly an H_s-valued random variable. For the case H_s is a complex Hilbert space there is a difficulty which we get around by requiring that H_s actually be a complex L_2-space. A complex white noise $w = w_1 + jw_2$ can then be defined to have the stated properties in terms of independent real and imaginary parts w_1 and w_2, each defined as above. The important point is that when projections are made onto finite-dimensional subspaces, w induces an ordinary white-noise vector.

Equation (2) is to provide the model for an estimation problem and it is desired that σ be estimatable. This means that $\mathcal{N}(B)$ is to be zero. This assumption is no more than a notational convenience, because if it is not originally satisfied one simply replaces H_σ with $\mathcal{N}(B)^\perp$. The white noise assumption is made because it seems appropriate in discussing attainable precision to assume as little *a priori* about what is being observed as possible in making an estimate. For the same reason, in choosing an estimation procedure to be applied we want one that requires no prior statistical information about σ. If the problem were finite-dimensional this condition would suggest the classical linear unbiased minimum variance (LUMV) estimator (see [1] or [5]). In fact, we are forced to finite-dimensional approximations, and we do use LUMV estimation. The case for using this estimator is strengthened by the facts that: (i) in the white-noise case it is the same as the least-squares estimator that minimizes observation error, and (ii) if w is assumed to be Gaussian, it is the same as the maximum-likelihood estimator.

There is a satisfactory and easy theory of linear least-squares estimation in Hilbert space, and a more complicated theory of LUMV estimation in Hilbert space. Neither can be used here, since the fact $\mathcal{R}(B)$ is not closed implies the estimators are unbounded operators, not everywhere defined; furthermore, for the LUMV estimator the existing theory requires the noise to be a proper H_s-valued random variable ([1]).

Linear Precision Guage

There are no doubt many reasonable ways to define "precision" so that it may be treated mathematically, and it is likely any definition will sometimes not be quite appropriate. However the notion to be introduced is flexible, in that it allows for various specific measures of precision, and it involves only linear spaces and operations and L_2-norms.

The simple idea involved is the following. A sequence of linear finite-dimensional approximations is set up such that at each stage a better approximation is given to the original problem than is given by its predecessor. At each stage an LUMV estimate is obtained. Early on in the sequence the detail or "precision" that is obtained is low because the approximation is rough; but the mean-squared error is relatively small. Thus one can get a low quality estimate, that is, however, statistically satisfactory. As one proceeds in the sequence of approximations the detail improves but also the mean-squared error increases. A threshold is set for tolerable mean-squared error, and the detail that is achieved when this is reached is thought of as the attainable precision; a proper definition follows below. Obviously, the attainable precision depends on signal-to-noise ratio and on the threshold. We allow the sequence of approximations to be more-or-less arbitrary. This allows one to design the sequence so that the increasing detail emphasizes, to some degree at least, the features desired. Thus we are talking about a whole family of precision measures. How well one can do with any of these is bounded from above by one particular sequence of approximations derived from the spectral decomposition of B^*B. This idea is now formalized.

Let N_k, $k=1,2,\ldots$, be an arbitrarily chosen sequence of subspaces of H_σ of finite dimension $d_k = \dim(N_k)$, such that $N_k \subset N_{k+1}$, $N_k \neq N_{k+1}$ and $Q_k x \to x$ as $k \to \infty$ for every $x \in H_\sigma$, where Q_k is the orthogonal projection on N_k. Define M_k in H_s to be $\mathcal{R}(BQ_k)$; since $\mathcal{R}(BQ_k)$ is finite-dimensional it is closed and since BQ_k is 1:1, $\dim(M_k) = d_k$. Let P_k be the orthogonal projection on M_k. If M denotes $\overline{\mathcal{R}(B)}$ in H_s and P is the orthogonal projection on M, then $P_k y \to Py$ for every $y \in H_s$. There now can be formulated a sequence of finite-dimensional estimation problems: given the observation $P_k z$ in

$$P_k z = P_k B\sigma + P_k w, \tag{3}$$

estimate $\sigma_k \triangleq Q_k \sigma$. Equation (3) can be rewritten

$$\begin{aligned} z_k &\triangleq P_k z = P_k B(Q_k\sigma + Q_k^{\perp}\sigma) + w_k \\ &= B\sigma_k + P_k B\, Q_k^{\perp}\sigma + w_k \end{aligned} \tag{4}$$

where $w_j \triangleq P_k w$ and $\sigma_k \triangleq Q_k\sigma$. If $\sigma \in N_k$, Eq. (4) reduces to

$$z_k = B\sigma_k + w_k = B_k\sigma_k + w_k, \tag{5}$$

where $B_k \triangleq P_k\, b\, Q_k$ can be interpreted as a transformation from N_k onto M_k. With this interpretation of domain and range B_k is invertible, so the LUMV estimate for σ_k in Eq. (5) is given simply by

$$\hat{\sigma}_k = B_k^{-1} z_k \tag{6}$$

The variance of the estimate (= mean-squared error since the estimate is unbiased) is $c\cdot\mathrm{Tr}[(B_k^*B_k)^{-1}]$. We define a normalized average variance per component,

$$e_k \triangleq \frac{1}{cd_k}\,\mathrm{var}\,\hat{\sigma}_k = \frac{1}{d_k}\,\mathrm{Tr}[(B_k^*B_k)^{-1}], \tag{7}$$

and call it the *error number*.

Definition. A *linear precision guage* (LPG) is a sequence $\{N_k\}_{k=1}^{\infty}$ of subspaces of H_σ satisfying the conditions stated above. A *level of precision* $\underline{n}$ is attainable with respect to the LPG $\{N_k\}$ and the threshold $\alpha > 0$ if n is such that $e_n \le \alpha$ and $e_k > \alpha$ for all $k > n$. If $e_k \le e_{k+1}$ for all k, the LPG is said to be *regular*.

For a given LPG $\{N_k\}$, when the estimator B_k^{-1} is applied to actual observed data z_k as given by Eq. (4) there is an error term $B_k^{-1}P_kBQ_k^{\perp}\sigma$. This error term is called the *offset* at the k-th level.

From a theoretical point of view the most important LPG is obtained from the spectral decompositions of B^*B and BB^*. We have

$$\begin{aligned} B^*B\,\psi_n &= \lambda_n\psi_n \\ BB^*\,\phi_n &= \lambda_n\phi_n \\ B\,\psi_n &= \lambda_n^{1/2}\phi_n \\ B^*\,\phi_n &= \lambda_n^{1/2}\psi_n \end{aligned} \tag{8}$$

where $\lambda_n > 0$, $\lambda_n \ge \lambda_{n+1}$, $\lambda_n \to 0$ as $n \to \infty$; and where $\{\psi_n\}_{n=1}^{\infty}$ is a c.o.n.s. in H_σ (since $\mathcal{N}(B) = 0$), and where $\{\phi_n\}_{n=1}^{\infty}$ is an o.n.s. in H_s that spans $\underline{\mathcal{R}(B)} = \underline{\mathcal{R}(BB^*)}$. Take N_k^o to be the subspace of H_σ spanned by $\{\psi_1,\ldots,\psi_k\}$; then M_k^o is the subspace of H_s spanned by $\{\phi_1,\ldots,\phi_k\}$. Obviously, $d_k = k$. We denote the orthogonal projections on N_k^o and M_k^o by Q_k^o and P_k^o, respectively. The error number for N_o^k is

$$e_k^o = \frac{1}{k} \sum_{n=1}^{k} \frac{1}{\lambda_n} \tag{9}$$

since the eigenvalues for $(B^*B)^{-1}$ are $\{\lambda_n^{-1}\}_{n=1}^{\infty}$. Since $\lambda_n \to 0$ as $n \to \infty$ it follows easily that $e_k^o \to +\infty$ monotonically as $k \to \infty$. Thus for any $\alpha > \lambda_1^{-1}$ a nonzero level of precision is attainable with respect to the LPG $\{N_k^o\}$. Since $P_k^o B\, Q_k^{o\perp} = 0$, the offset is zero at all levels for any σ. We state a Proposition that summarizes these facts and also indicates the role of the LPG $\{N_k^o\}$ as a bound for any LPG.

Proposition.

(1) There exists an LPG $\{N_k^o\}_{k=1}^{\infty}$ with error numbers e_k^o with the following properties: (i) $d_k = k$; (ii) $\{N_k^o\}$ is regular; (iii) $\{N_k^o\}$ has zero offset.

(2) If $\{N_k\}_{k=1}^{\infty}$ is any LPG, with error numbers e_n, and $\dim N_n = d_n$, then $e_n \geq e_{d_n}^o$.

(3) For any LPG $\{N_k\}_{k=1}^{\infty}$ and any threshold α greater than some minimum, there is a finite attainable level of precision.

(4) An LPG $\{N_k\}_{k=1}^{\infty}$ has zero offset at all levels iff the N_k are direct sums of eigenspaces of B^*B.

The assertion (1) has been shown; (3) follows immediately from (1) and (2). The proofs of (2) and (4) are straightforward but are not given.

Remarks.

(1) The noise intensity c is divided out in the definition of e_k because its effect can be accounted for by changing α; in fact $d_k^{-1} \operatorname{var} \hat{\sigma}_k > \alpha$ is the same as $e_k > c^{-1}\alpha$.

(2) In general an offset term is present and causes an error not accounted for in the error number e_k; e_k is proportional to the mean-squared error only in the idealized case that $\sigma \in N_k$. Let $\|B\|_k$ denote the norm of B when restricted to $N_k^{\perp}$. Then

$$\text{offset error} = \|B_k^{-1} P_k B\, Q_k^{\perp}\sigma\| \leq \|B_k^{-1}\| \cdot \|B\|_k \|\sigma_k\| \tag{10}$$

Unfortunately, this simple bound cannot be improved upon without additional information (as is the case with $\{N_k^o\}$), or without constraints on σ. Thus the "attainable level of precision," while always giving an upper bound, can often only nearly be attained for σ that lie close to N_k.

(3) Thus far the observation space H_s has been taken for granted. Often there appears to be a choice available, and some comment is in order. Implicit in the interpretation of e_n are two extra-mathematical assumptions: (i) H_s contains all available experimental data, and (ii) there is no information available that would permit one to improve on the

supposition that the observations contain irreducible white noise at a level of intensity c. If H_{s1} is a space that satisfies these conditions and G is a mapping from H_{s1} into a space H_{s2} (a prefilter), it is not "fair" to take H_{s2} as the observation space unless G is a linear transformation that is 1:1 from $\mathcal{R}(B)$ onto a subspace of H_{s2}, and G is isometric. Then nothing is lost in moving to H_{s2}, and the white noise assumption is equally valid there.

Sometimes, however, notably when it is desired to consider sampled data, there is good reason to replace the ideal $H_s = H_{s1}$ with a restricted observation space H_{s2} which is only finite-dimensional. The mapping B is then replaced by $GB{:}H_\sigma \rightarrow H_{s2}$, which does have closed range, and the new problem based on the model

$$z' \;=\; GB\sigma + Gw \tag{11}$$

is not ill-posed, technically speaking. But it will be "practically ill-posed" if H_{s2} is of sufficiently large dimension. Everything goes much as before, but with these changes: H_σ must be replaced with the finite-dimensional subspace $\mathcal{N}(GB)^\perp$ so that the N_k are replaced with $N_k \cap \mathcal{N}(GB)^\perp$, and e_n does not approach infinity; it merely takes on unacceptably high values. In the example discussed below, H_{s2} is a restricted observation space whose elements are sequences of sampled values taken as densely as possible.

A FOURIER IMAGING PROBLEM

There is space here only to sketch this example; for details see [6]. With $\sigma \in L_2$, let

$$s(\xi) \;=\; I_{[-b,b]}(\xi)\int_{-a}^{a} e^{-j2\pi\xi x}\sigma(x)\,dx \tag{12}$$

where $I_{[-b,b]}$ is the indicator function for the interval indicated. The operator B, $B\sigma = s$, defined by the equation is Hilbert-Schmidt. Furthermore, s is uniformly continuous for $-b \leq \xi \leq b$ (and equicontinuous for $\|\sigma\| \leq$ constant) so it makes sense to consider sampled values of s. In order to make things fit nicely in what is to follow it is assumed $\mu \triangleq 4ab =$ even integer.

An LPG $\{N_k\}$ is defined as follows. First the interval [-a,a] is partitioned into $\mu_1 \triangleq \mu$ intervals or cells of length 1/2b. Put

$$\sigma_k^{(1)} \;\triangleq\; 2b\int_{k/2b}^{(k+1)/2b} \sigma(x)\,dx, \quad k = -K_1, -K_1+1, \ldots, K_1-1 \tag{13}$$

where $K_1 \triangleq \mu_1/2$. Define $\eta_k^{(1)}(x) \triangleq I_{[k/2b,(k+1)/2b)}(x)\sigma_k^{(1)}$, and

$$\sigma^{(1)}(x) \triangleq \sum_{k=-K_1}^{K_1-1} \eta_k^{(1)}(x). \qquad (14)$$

Thus $\sigma^{(1)}$ is the orthogonal projection of σ into the μ_1-dimensional subspace comprised of all functions that are constant a.e. on the cells of the partition. This subspace is N_1, so $d_1 = \mu_1 = \mu$. To define N_2, the interval $[-a,a]$ is partitioned into $d_2 = 2d_1$ intervals of length $1/4b$, and the above construction is mimicked to give $\sigma^{(2)}$ which takes on constant values on the smaller cells. This construction is continued, with the cells of the previous partition being split in half each time, to yield a sequence of subspaces $\{N_n\}$. It follows that $N_n \subset N_{n+1}$ and Q_n converges strongly to the identity, so $\{N_n\}$ is an LPG. The notations d_n, $\eta_k^{(n)}$ and $\sigma^{(n)}$ are defined in the obvious way; $d_n = 2^{n-1}\mu$. We also put $K_n \triangleq d_n/2$, $a_n \triangleq 2^{n-1}a$, $b_n \triangleq 2^{n-1}b$ and $\mu_n \triangleq (2^{n-1})^2\mu$.

It is known that $\mathcal{R}(B) = L_2[-b,b]$ so that the natural H_s for this problem is $L_2[-b,b]$. The subspace M_m consists of all functions of the form

$$s(\xi) = \frac{1}{2b} e^{j\pi\xi/2b_n} \frac{\sin(\pi\xi/2b_n)}{\pi\xi/2b_n} \sum_{k=-K_n}^{K_n-1} \sigma_k^{(n)} e^{j2\pi k\xi/2b_n}, \qquad (15)$$

but this is not used here, since $L_2[-b,b]$ is to be replaced by a restricted finite-dimensional observation space using sampled data, as mentioned in the remark above. For any positive integer n let $\tilde{M}_n$ be the d_n-dimensional linear space of sampled values of $s(\xi)$ at the points $(m/2a_n)$, $m=-K_n,\ldots,K_{n-1}$. As a convenience, let $\theta^{-1}(\mu) \triangleq e^{ju}\frac{\sin u}{u}$. Then, from Eq. (15),

$$q_m^{(n)} \triangleq \theta(\pi m/\mu_n)s(m/2a_n) = \frac{1}{2b}\sum_{k=-K_n}^{K_n-1} \sigma_k^{(n)} e^{j2\pi mk/\mu_n} \qquad (16)$$

If we suppose the sampled values $s(m/2a_n)$ are replaced by averages over very short intervals (much shorter than the Nyquist intervals of length $1/2a_\nu$), then the mapping from $L_2[-b,b]$ to $\tilde{M}_n$ is an orthogonal projection G_n, and a vector $\underline{s}^{(n)} \in \tilde{M}_n$ is given by $\bar{s}^{(n)} = G_n B_n \sigma^{(n)}$. According to Eq. (16), $\underline{s}^n$ is also given by a truncated dFt (discrete Fourier transform) followed by a multiplication operator (approximately, since Eq. (16) involves actual sampled values — but we neglect the error, which may be made as small as desired if $\|\sigma\|$ is bounded). We write $\underline{s}^n = A_n \Delta_n$, where Δ_n is the truncated dFt and A_n is the multiplication.

Now take $H_{s2} = \tilde{M}_\nu$, ν a positive integer, and consider the error numbers $e_n^{(\nu)}$, $1 \le n \le \nu$, for the LPG $\{N_n\}$ and this observation space. First take $n = \nu$. Let ρ_ν be the $2^{\nu-1}\mu \times 2^{\nu-1}\mu$ - matrix $[\rho_\nu(n-m)]$, where

$$\rho_\nu(n-m) \triangleq \frac{\sin(\pi(n-m)/2^{\nu-1})}{\sin(\pi(n-m)/\mu\nu)}, \tag{17}$$

$n,m = -K_\nu,\ldots,K_\nu-1$ (recall that $2K_\nu = d_\nu = 2^{\nu-1}$). Let

$$\gamma_\nu \triangleq \left|\frac{\sin(\pi K_\nu/\mu_\nu)}{\pi K_\nu/\mu_\nu}\right|,$$

which equals $(2/\pi)$ for $\nu=1$ and approaches 1 as ν becomes larger. Then it follows from Eq. (7) that

$$\frac{1}{2^{\nu-1}\mu}\,\mathrm{Tr}[\rho_\nu^{-1}] \le e_\nu^{(\nu)} \le \frac{1}{2^{\nu-1}\gamma_\nu^2\mu}\,\mathrm{Tr}[\rho_\nu^{-1}]. \tag{18}$$

The left side of this inequality equals 1 when $\nu=1$, which corresponds to sampling at the Nyquist rate. The left side of (18) grows rapidly with ν as one would expect (some representative values are calculated in [6]).

When $1 \le n < \nu$, the matrix trace that has to be calculated to give $e_n^{(\nu)}$ does not appear to reduce to anything very simple and we do not give the formula here. One can see intuitively, however, and there is an easy pseudo-proof [6], that

$$e_n^{(\nu)} \le \frac{1}{2^{\nu-n}}\, e_n^{(n)} \tag{20}$$

(the "pseudo-proof" involves an approximation, but one that is very good for reasonably large values of μ).

Remarks.

(1) This brief account does not bring out the role the particular white noise assumption made plays in the normalization of the norm in $\tilde{M}_n$ as n changes. It is assumed that the noise variable added to each sampled value has fixed variance c irrespective of the sampling rate (see [6]).

(2) This example can reasily be extended to higher spatial dimension than one — up to the point where it is desired to carry out computations.

(3) The fact that precision is limited in the Fourier imaging problem no matter what data processing is used, which follows from the Proposition stated earlier, is already implied by the results of the classic paper [7].

REFERENCES

[1] Beutler, F. J. and Root, W. L. (1976), "The Operator Pseudo-Inverse in Control and Systems Identification," in Generalized Inverses and Applications, ed. Nashed, pp. 397-494, Academic Press.

[2] Walker, J. L. (1980). "Range-Doppler Imaging of Rotating Objects," IEEE Trans. on Aero. and Electronic Systems, v AEC-16, n1, pp. 23-52.

[3] Brown, W. M. (1980). "Walker Model for Radar Sensing of Rigid Target Fields," IEEE Trans. on Aero and Electronic Systems, v AES-16, n1, pp. 104-106.

[4] Berman, N. and Root, W. L. (1981). "On Stochastic Linear Differential Equations in Reflexive Banach Spaces," Proc. 1981 International Symposium on the Math. Theory of Networks and Systems, Santa Monica, California.

[5] Albert, A. (1972), Regression and the Moore-Penrose Pseudoinverse. Academic Press, New York.

[6] Joyce, L. S. and Root, W. L. (1982). "Precision and Ill-Posedness in Linear Imaging Problems," Technical Report. Environmental Research Institute of Michigan, Ann Arbor.

[7] Pollak, M. O. and Slepian, D. (1961). "Prolate Spheroidal Wave Functions, Fourier Analysis and Uncertainty. I," Bell Systems Tech. J., v. 40, pp. 65-84.

ACKNOWLEDGEMENT

The research of the second-named author was supported by the National Science Foundation under Grant ECS-8005960.

A SIMPLE NONLINEAR EDGE PRESERVING FILTER

CARLOS A. POMALAZA
NIHE, Dep. of Electronics
Limerick, IRELAND

CLARE D. MCGILLEM
Purdue University, School of Elec. Eng.
West Lafayette, IN 47907 USA

INTRODUCTION

The use of nonlinear filters in signal processing has been increasing in recent years. One of the most popular nonlinear filters is the median filter suggested initially by Tukey [1]. One of the main features of this kind of filter is that under certain conditions [2] it preserves the edges present in the signal. It is because of this property that it has been found useful in image processing [3], [4]. Exact computation of the mean square errors (MSE) by Justusson [5], shows that running median filters are better (in the MSE sense) than linear averaging when moving *across* the edges but are worse when moving *along* the edges; i.e., the uniform sections on each side of the edge. For many applications in which these filters are employed it is the signal that provides the "jumps" in the data (the edges) and the noise is relatively well behaved; i.e., the noise has a probability density function (PDF) with fast decaying tails. An example of this is the quantization noise in digital image processing. It is for situations such as these that the filter proposed here has been found to have better performance than linear averaging or median filtering. Furthermore, the computation doesn't require order statistics of the data in the filter window and in certain cases it can be as simple as linear smoothing. It will be also shown that the use of this filter for impulse type signals provides a substantial improvement when compared to other methods that either smooth or distort the impulse.

THE FILTER

The idea behind the formulation of the filter is that of having a linear filter whose window dimensions adjust automatically, depending on the nature of the signal being processed; i.e., whether it is an edge or a smooth region. A filter of the following form incorporates these ideas

$$\hat{s}_i = \frac{\sum_{j=1}^{N} w(r_j - \tilde{s}_i) r_j}{\sum_{j=1}^{N} w(r_j - \tilde{s}_i)} \tag{1}$$

where

r_j are the measured data samples
$\tilde{s}_i$ is an estimate of the signal sample s_i
$w(\cdot)$ is a weighting function

$\hat{s}_i$ is the estimate provided by the filter

Equation (1) resembles the M-type estimators used in robust statistics [6]. The application here is different from robust statistics, however, where it is the noise that provides the sudden variations in the amplitude of the measured data. Most of the available results about estimators of this type are in terms of their asymptotic properties. A computational difficulty arises when it is desired to evaluate the *exact* performance, (MSE), for N finite. For this case the preferred method used in the calculation of the variance or MSE has been Monte-Carlo computer simulations [7]. These difficulties are increased for estimates as in (1) where, although the samples r_j can still be assumed to be independent, they are not necessarily identically distributed. The following simple form of w(x), used in this paper, allows one to compute exactly the filter performance (MSE) by the procedure shown in the APPENDIX.

$$w(x) = \begin{cases} 1 & \text{for } |x| \leq c \\ 0 & |x| > c \end{cases} \tag{2}$$

and instead of $\tilde{s}_i$ the actual measured data sample r_i will be used. The value of c in (2) is chosen empirically and depends on the variance of the contaminating noise. It is apparent from (1) and (2) that with no noise present and with an edge of height h, for any value of c between 0 and h the filter output sequence will be exactly the same as the input sequence; i.e., the edge will be preserved.

In order to compare the results using different processors consider the following model. The measured samples are represented by

$$r_i = s_i + n_i \tag{3}$$

where

n_i is the noise sample

s_i is the signal sample near an edge; i.e., $s_i = s$ for $i = 1,...k$ and $s_i = s + h$ for $i = k + 1,...$

r_i can be considered as an independent random sample of two populations, $r_1, r_2,... r_k \sim f_1(r)$ and $r_{k+1}, r_{k+2},... \sim f_2(r)$. In the application here the PDF's $f_1(r)$ and $f_2(r)$ are of the same form but of different mean value.

To measure the performance of the filters considered in this paper the sum of the MSE's of the filter output sequence will be used. For a filter of length N there are N-1 estimates, $\hat{s}_i$, since the filter window encounters and edge until it leaves it, as shown in Figure 1.

The MSE of the sequence $\{\hat{s}_i\}$ is defined as

$$\text{MSE}\{\hat{s}_i\} = \sum_{i=1}^{N-1} E(\hat{s}_i - s_i)^2 \tag{4}$$

Because the filters considered in this paper are location invariant and in view of the symmetry of the problem, (4) can be written as

$$\text{MSE}\{\hat{s}_i\} = 2 \sum_{i=1}^{(N-1)/2} E(\hat{s}_i)^2 \tag{5}$$

When the noise is Gaussian the MSE of $\{\hat{s}_i\}$ as in (1), with the assumptions of (2),

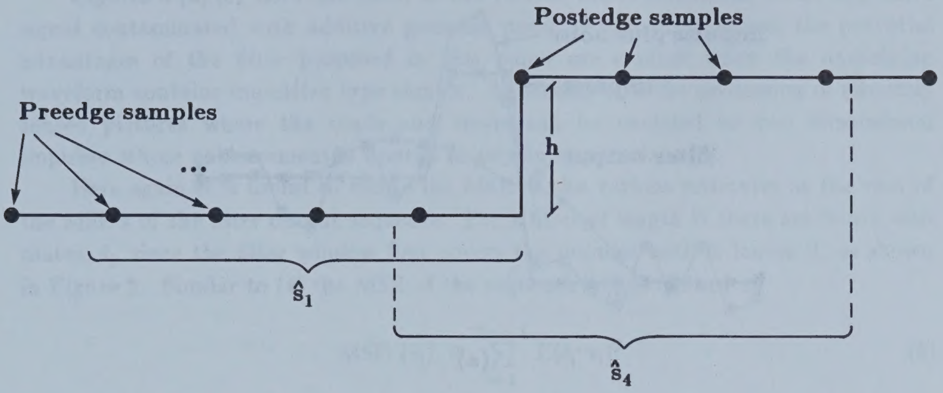


Figure 1. Filter output sequence $\{\hat{s}_i\}$, N = 5.

can be exactly evaluated for various values of N, h, and c (see APPENDIX). This MSE can then be compared with the values obtained using the running mean and running median filters.

Extension to the filtering of an impulse of width n.

An impulsive signal of width n and steps h_1 and h_2 is defined here as in Figure 2. When this type of signal is subjected to linear filtering a certain degree of smoothing always results. In the case of nonlinear processors like the median filter, depending on the filter length and the impulse width the result can, in certain cases, be a severe distortion or translation of the original signal [2].

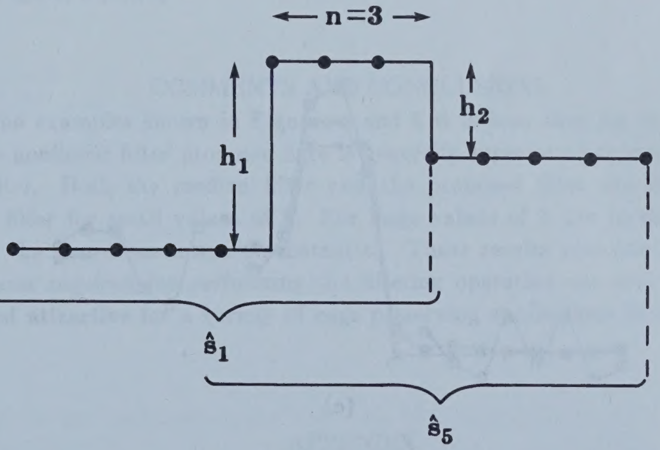


Figure 2. Filter output sequence $\{\hat{s}_i\}$, N = 9, n = 3.

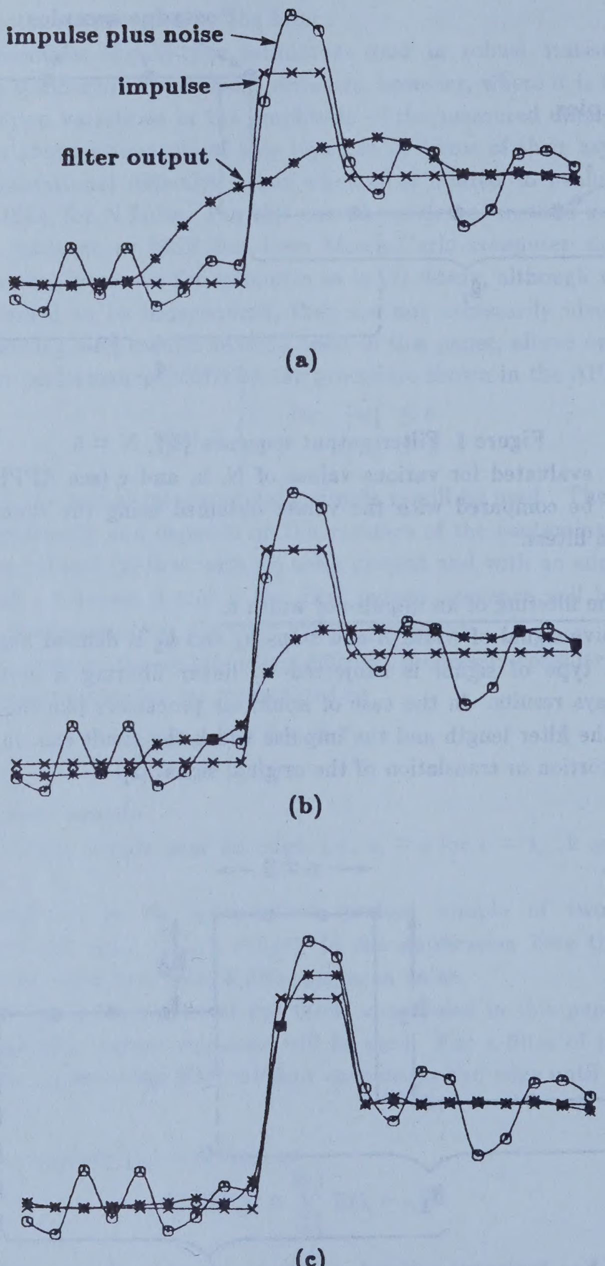


Figure 3. Results of filtering an impulse of n=3, h_1=6.0, h_2=3.0 contaminated with additive gaussian noise (σ^2=1.0). (a) running mean, (b) running median, (c) as in Eq. (1). The filter length, N, for all cases is 9.

Figures 3 (a)-(c) show the effect of the various filters mentioned on an impulsive signal contaminated with additive gaussian noise. From these figures the potential advantages of the filter proposed in this paper are evident when the underlying waveform contains impulsive type signals. An example, is the processing of remotely sensed pictures where the roads and rivers can be modeled as two dimensional impulses whose enhancement is desired in certain applications [8].

Here again it is useful to define the MSE of the various estimates as the sum of the MSE's of the filter output sequence. For a filter of length N there are N-n-1 estimates, $\hat{s}_i$, since the filter window first covers the impulse until it leaves it, as shown in Figure 2. Similar to (4) the MSE of the sequence $\{\hat{s}_i\}$ is defined as

$$\text{MSE}\ \{\hat{s}_i\} = \sum_{i=1}^{N-n-1} E(\hat{s}_i - s_i)^2 \tag{6}$$

The situations where the filter window only covers one side of the impulse can be considered as cases of the edge filtering problem discussed previously. By methods explained in the APPENDIX the MSE $\{\hat{s}_i\}$ of a filter as in (1) can be exactly computed and compared with the values for the running mean and running median filters.

NUMERICAL RESULTS

Figures 4 (a)-(d) show the various results obtained by using the equations from the APPENDIX for the edge filtering problem. The contaminating noise is assumed to be gaussian, zero mean with $\sigma^2=1$. For different values of σ there is no need to recompute the results shown but only to scale the height of the edge h and the constant c; i.e., $h^1=h/\sigma$, $c^1=c/\sigma$. Similar results for the problem of impulse filtering are shown in Figures 5 (a)-(b).

COMMENTS AND CONCLUSIONS

For the examples shown in Figures 4 and 5 it is seen that (in the MSE sense) the simple nonlinear filter proposed here is generally superior when compared to the median filter. Both the median filter and the proposed filter are inferior to the averaging filter for small values of h. For large values of h the increase in performance by the new filter is quite substantial. These results plus the fact that the computations required for performing the filtering operation are very simple make the method attractive for a variety of edge preserving applications in signal processing.

APPENDIX

Assume that there is a random sample of independent random variables $r_1, r_2, \ldots r_N$. $r_1, r_2, \cdots r_m$ are distributed as $f_1(n)$, and $r_{m+1}, \ldots r_{m+n}$ are distributed as $f_2(r)$. For the application here $f_1(r) = f(r-s_0)$ and $f_2(r) = f(r-s_0-h)$, where f(r) is the

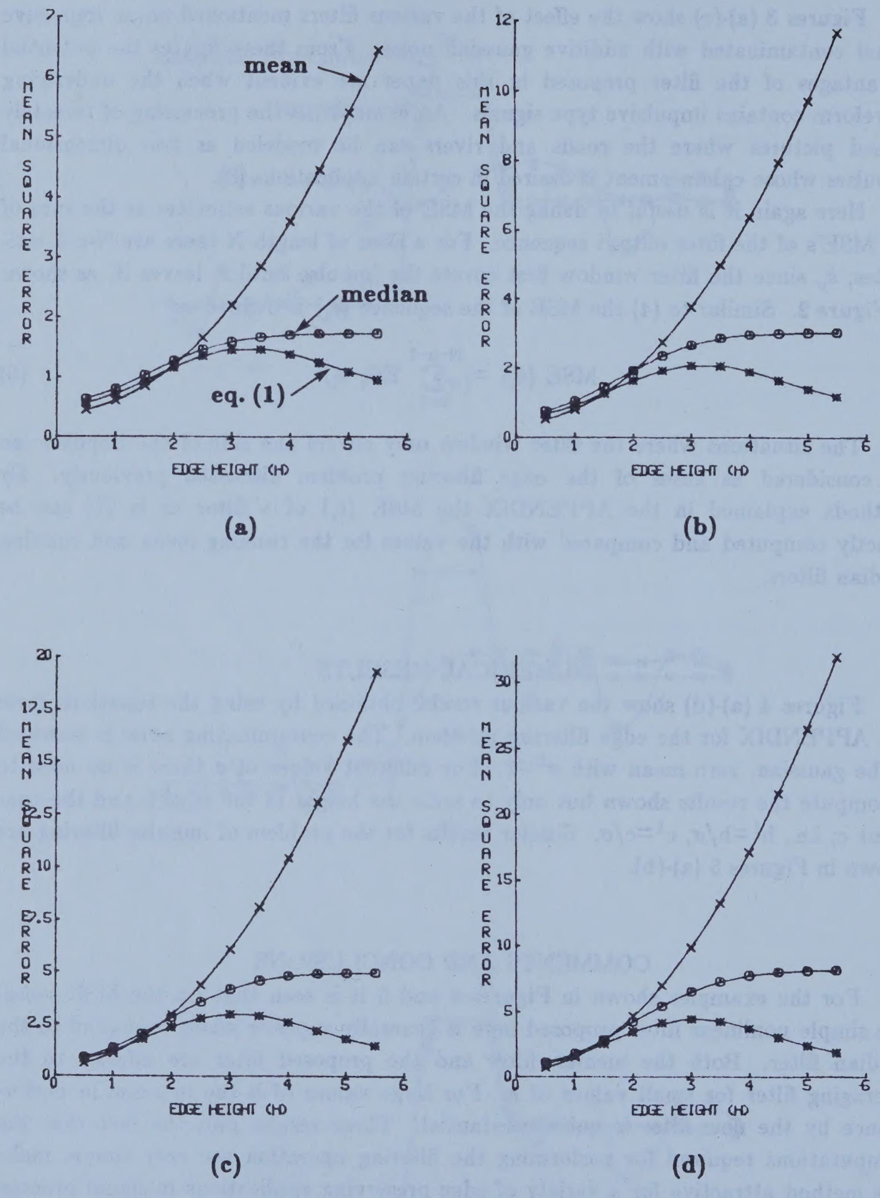
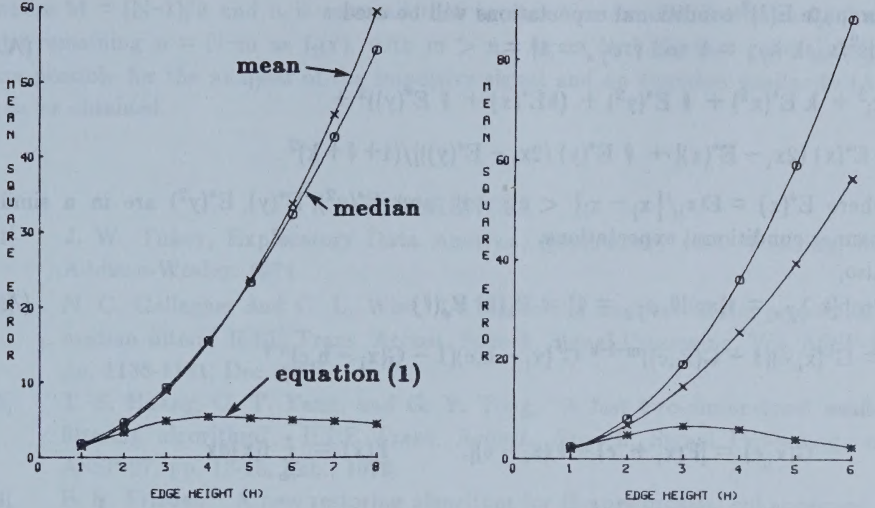


Figure 4. MSE comparison for the various filters discussed. (a) N=5, (b) N=9, (c) N=15, (d) N=25. For all cases the value of c in (2) is 3.0.



(a) (b)

Figure 5. MSE comparison, impulse filtering. ($c=3.0$) (a) $N=9$, $n=3$, $h_1=h$, $h_2=h/2$ (b) $N=9$, $n=3$, $h_1=h$ $h_2=h$.

noise distribution, s_0 is the base signal amplitude and h is the edge height.

It is desired to compute the MSE of the estimate of s given by

$$\hat{s} = \frac{\sum\limits_{j=1}^{N} w(r_j - r_i) r_j}{\sum\limits_{j=1}^{N} w(r_j - r_i)} \tag{A.1}$$

where $w(\cdot)$ is as in (2). In (A.1) it is assumed that N is odd, $i = (N-1)/2$ and $m > n$. Renaming $r_1, r_2, \ldots r_m$ as $x_1, x_2, \ldots x_m$ and $r_{m+1}, \ldots r_{m+n}$ as $y_1, y_2, \ldots y_n$ (A.1) can be written as

$$\hat{s} = \frac{\sum\limits_{j=1,\neq i}^{m} \lambda_j x_j + \sum\limits_{j=1}^{n} \phi_j y_j + x_i}{1 + \sum\limits_{j=1,\neq i}^{m} \lambda_j + \sum\limits_{j=1}^{n} \phi_j} \tag{A.2}$$

where

$$\lambda_j = \begin{cases} 1 & \text{if} \left| x_j - x_i \right| < c \\ 0 & \text{otherwise} \end{cases}$$

$$\phi_j = \begin{cases} 1 & \text{if} \left| y_j - x_i \right| < c \\ 0 & \text{otherwise} \end{cases}$$

Without loss of generality assume $s_0=0$, then MSE($\hat{s}$) is equal to $E(\hat{s})^2$. In order to compute $E(\hat{s})^2$ conditional expectations will be used.

$$E[\hat{s}^2/x_i,\ k\ \lambda_{j's} = 1 \text{ and } \ell\ \phi_{j's} = 1] = \tag{A.3}$$

$$[x_i^2 + k\,E^*(x^2) + \ell\,E^*(y^2) + (kE^*(x) + \ell\,E^*(y))^2 +$$

$$k\,E^*(x)\,(2x_i - E^*(x)] + \ell\,E^*(y)\,(2x_i - E^*(y))]/(1+\ell+k)^2$$

where $E^*(x) = E(x_j/|x_j - x_i| < c, j \neq i]$, and $E^*(x^2)$, $E^*(y)$, $E^*(y^2)$ are in a similar manner conditional expectations.

Also,

$$\text{Prob}[k\ \lambda_{j's} = 1]\text{and}\ell\ \phi_{j's} = 1] = P_\lambda(k)\,P_\phi(\ell) \tag{A.4}$$

$$= G^k(x_i,c)(1 - G(x_i,c))^{m-l-k}\,G^\ell(x_i - h,c)(1 - G(x_i - h,c)^{n-\ell}$$

where

$$G(x_{i,}c) = [F(x_i + c) - F(x_i - c)], \qquad F(x) = \int_{-\infty}^{x} f(x)dx$$

$$E[\hat{s}^2/x_i] = \sum_{k=0}^{m-1} \begin{pmatrix} m-1 \\ k \end{pmatrix} P_\lambda(k) \left[\sum_{\ell=0}^{n} \begin{pmatrix} n \\ \ell \end{pmatrix} P_\phi(\ell)\, E(\hat{s}^2/x_{i,}\ k\ \lambda_{j's} = 1 \text{ and } \ell\ \phi_{j's} = 1 \right] \tag{A.5}$$

Finally

$$E(\hat{s}^2) = \int E(\hat{s}^2/x_i)f(x_i)dx_i \tag{A.6}$$

For $f(x) = N(0, \sigma^2)$ all but (A.6) can be computed as closed-form expressions. For (A.6) a Gaussian quadrature numerical integration can be used.

An observation note must be included here regarding the value of the parameter c, used for the results shown in Figure 4. It is always possible to carry out a numerical minimization of (A.6) for a fixed value of h. However, in this paper, the value of $c = 3$ has been found, by trying out several other values, to be very close to providing an *overall* minimum MSE for the various values of h used. More formal optimization methods such as minimax analysis could be applied here, but the numerical complexities can be formidable.

In the case of an impulsive signal the random variables $r_1, r_2, \ldots r_N$ can be assumed to be divided in three groups: $x_1, x_2, \ldots x_m$ distributed as $f_1(x)$, $y_1, y_2, \ldots y_n$ distributed as $f_2(y)$ and $z_1, z_2, \ldots z_p$ distributed as $f_3(z)$, where $m+n+p=N$ and n is the impulse width. By analysis similar to that used for the edge signal, equations of the form (A.3)-(A.5) can be derived and used for the MSE calculations.

For the median filter MSE calculations, an extension of the methods used for obtaining the PDF of the median of a random i.i.d. sample gives the following PDF of the median of the independent random sample from two populations [5].

$$g(x) = mf_1(x) \sum_{j=M-n}^{M} \begin{pmatrix} m-1 \\ j \end{pmatrix} \begin{pmatrix} n \\ M-j \end{pmatrix} F_{1(x)}^{j} F_{2(x)}^{M-j} [1-F_1(x)]^{m-1-j} [1-F_2(x)]^{n-M+j}$$

$$+ nf_2(x) \sum_{j=M-n+1}^{M} \begin{pmatrix} M \\ j \end{pmatrix} \begin{pmatrix} n-1 \\ M-j \end{pmatrix} F_1^j(x) F_2^{M-j} [1-F_1(x)]^{m-j} [1-F_2(x)]^{n-1-M+j} \tag{A.7}$$

where $M = (N-1)/2$ and it is assumed that m variables are distributed as $f_1(x)$ and the remaining $n = N-m$ as $f_2(x)$, with $m > n$. Again here, further generalizations are possible for the analysis of the impulsive signal and an equation similar to (A.7) can be obtained.

REFERENCES

[1] J. W. Tukey, Exploratory Data Analysis. (preliminary ed.). Reading, MA: Addison-Wesley, 1971.

[2] N. C. Gallagher and G. L. Wise, "A theoretical analysis of the properties of median filters" IEEE Trans. Acoust. Speech, Signal Processing, Vol. ASSP-29, pp. 1136-1141, Dec. 1981.

[3] T. S. Huang, G. T. Yang, and G. Y. Tong, "A fast two-dimensional median filtering algorithm". IEEE Trans. Acoust., Speech, Signal Processing, vol. ASSP-27, pp. 13-18, Feb., 1979.

[4] B. R. Frieden. "A new restoring algorithm for the preferential enhancement of edge gradients" J. Opt. Soc. Amer., vol. 66, pp. 280-283, 1976.

[5] B. Justusson, "Median Filtering: Statistical Properties," Roy. Inst. Technol., Stocklom, Sweden, Tech. Rep. TRITA-MAT-1979-11 (August).

[6] P. J. Huber, "Robust Statistical Procedures" SIAM. Regional Conference Series in Applied Mathematics, No. 27, 1977.

[7] D. F. Andrews, P. J. Bickel, F. R. Hampel, P. J. Huber, W. H. Rogers, J. W. Tukey, Robust Estimates of Location: Survey and Advances, Princeton University Press, N.J., 1972.

[8] P. E. Anuta, C. A. Pomalaza, F. Davallou, "Comparison of Edge Detection Methods for Landsat Imagery," Symposium Proceedings, Machine Processing of Remotely Sensed Data, Purdue University, West Lafayette, June 1982.

EDGE DETECTION USING CONTOUR TRACING[†]

EDWARD J. DELP and CHEE-HUNG HENRY CHU
Department of Electrical and Computer Engineering
and
Computer, Information and Control Engineering Program
University of Michigan
Ann Arbor, Michigan 48109

ABSTRACT

An approach to edge detection using the direction information provided by both the gradient masks and template masks operators is examined. The edge direction is used to trace an edge segment. Three edge operators developed using this approach are discussed. An evaluation of their performances compared to the Sobel operator indicates they produce better connected edge segments and thinner edges. Results of applying these edge detectors to real-world images are also presented.

1. Introduction

The fundamental step of image analysis is segmentation, which partitions the image into individual objects. One method of segmentation is gray level edge detection. The output of edge detectors are usually linked together to form continuous boundaries for further processing, such as shape analysis. Hence, besides the accuracy in edge location, desirable features of the output should also include thinness and continuity of edge segments.

One approach to the problem of edge detection is the parallel algorithms discussed by Davis [1]. One version of this parallel approach is enhancement/thresholding. The enhancement/thresholding algorithms use spatial operators to enhance the original image to form an edge enhancement strength map. The value of the strength map at each pixel is a function of the two-dimensional spatial convolution of the spatial operators and the image subarea in the neighborhood of that pixel. Typically, these spatial operators are 3 × 3 or 2 × 2 windows. The strength map contains values that are high at areas with large change in gray level and low at areas with almost constant gray level. A thresholding decision can then be applied to the strength map to determine the presence or absence of edge points.

Spatial operators can be differential operators (e.g. gradient-like operators) or template matching operators. Both types of operators provide a measure of the rate of increase/decrease of the gray level as well as the direction of the maximum increase/decrease. The direction information at any point can be used to determine the edge direction. Robinson [2] has shown that the edge direction information can be used to improve the performance of the enhancement/thresholding edge detectors.

[†] This research was partially sponsored by the Center for Robotics and Integrated Manufacturing, Robot Systems Division, College of Engineering, University of Michigan and the Air Force Office of Scientific Research under Grant F49620-82-C-0089.

This paper presents an edge detection algorithm that makes use of direction information for connectivity analysis. The basic scheme of the method is shown in Figure 1. The decision strategy classifies a pixel as an edge point according to its location, strength of the edge enhancement map, edge direction, and the gray level value at the pixel. The strength map is thresholded and connectivity analysis is used to trace an edge segment.

This segmentation-following-enhancement approach can be implemented as a sequential edge detector since the enhancement operation need not be applied to every spatial point.

Two connectivity strategies and three edge detectors are considered. An empirical evaluation of their performance under additive white Gaussian noise as well as the results of application of these edge operators is presented.

2. 3 x 3 Spatial Operators

The inner product of a 3 × 3 mask and 3 × 3 image subarea is defined as (the notation is due to Frei and Chen[3]):

$$(W,X) = \sum_{i=1}^{3}\sum_{j=1}^{3} \omega_{ij}\ x_{ij} \tag{1}$$

$$\textit{where} \quad W = \begin{bmatrix} \omega_{11} & \omega_{12} & \omega_{13} \\ \omega_{21} & \omega_{22} & \omega_{23} \\ \omega_{31} & \omega_{32} & \omega_{33} \end{bmatrix} \quad \textit{is the mask}$$

$$\textit{and} \quad X = \begin{bmatrix} x_{11} & x_{12} & x_{13} \\ x_{21} & x_{22} & x_{23} \\ x_{31} & x_{32} & x_{33} \end{bmatrix} \quad \textit{is the image subarea}$$

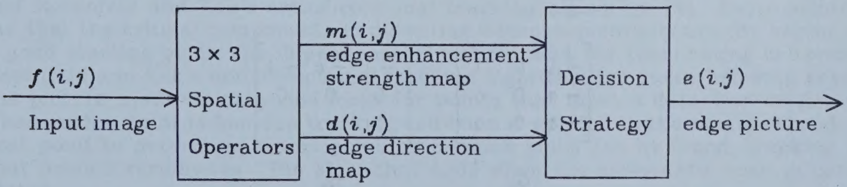


Figure 1. Overview of Proposed Edge Detection Algorithm

2.1. Differential Operators

These types of operators implement gradient-like function in discrete form and are of the form shown in Figure 2. Let X be the 3×3 subarea of $f(m,n)$ centered at (i,j). We can define discrete differentiation at pixel (i,j) in the y direction as:

$$(G_y,X) = f_x(i+1,j) - f_x(i-1,j) \tag{2}$$

where $f_x(m,n)$ is a weighted average of $f(m,n)$ in the x direction. The weight is determined by the value of c. Similarly, discrete differentiation in x direction is:

$$(G_x,X) = f_y(i,j+1) - f_y(i,j-1) \tag{3}$$

where $f_y(m,n)$ is a weighted average of $f(m,n)$ in the y direction.

The strength of the enhancement map is a multiple of the magnitude of the gradient.

$$m(i,j) \;=\; \sqrt{(G_x,X)^2 + (G_y,X)^2} \tag{4}$$

The direction value $\vartheta(i,j)$ is the direction of gradient.

$$\vartheta(i,j) = tan^{-1}\frac{(G_y,X)}{(G_x,X)} \tag{5}$$

The isotropic gradient masks that we used have $c = \sqrt{2}$.

2.2. Template Match Operators

These operators do not produce a gradient-like magnitude. Instead, they produce the discrete differentiation in eight specific directions from a set of 8 templates. The strength of the enhancement map is the maximum output value from these 8 templates.

$$m(i,j) = \max_{k}\ \{(W_k,X)\} \tag{6}$$

where W_k's are the template masks.

The direction value $\vartheta(i,j)$ corresponds to the direction associated with the largest output. The set that we have used is the 3-level template masks [2] as shown in Figure 3.

$$\underset{G_y}{\begin{bmatrix} -1 & 0 & 1 \\ -c & 0 & c \\ -1 & 0 & 1 \end{bmatrix}} \qquad \underset{G_x}{\begin{bmatrix} -1 & -c & -1 \\ 0 & 0 & 0 \\ 1 & c & 1 \end{bmatrix}}$$

Figure 2. Differential Gradient Operators

$$\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & 1 & 1 \\ -1 & 0 & 1 \\ -1 & -1 & 0 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix} \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & -1 \\ 0 & -1 & -1 \end{bmatrix}$$

$$W_0 \qquad W_1 \qquad W_2 \qquad W_3$$

$$\begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix} \begin{bmatrix} 0 & -1 & -1 \\ 1 & 0 & -1 \\ 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} -1 & -1 & 0 \\ -1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}$$

$$W_4 \qquad W_5 \qquad W_6 \qquad W_7$$

Figure 3. 3-level template masks.

3. Edge Detection by Contour Tracing

3.1. Algorithm

We shall assume that the edge direction at any point is perpendicular to the gradient direction. We define the edge direction $d(i,j)$ to be

$$d(i,j) = \vartheta(i,j) - \frac{\pi}{2}. \tag{7}$$

where $\vartheta(i,j)$ = direction value found by the spatial operator.

Depending on the connectivity scheme, $d(i,j)$ can be quantized to one of 8 values using chain codes.

In general terms, to be classified as an edge point, a point must be at the right spatial location relative to a previously found edge point, have an edge direction that provides good continuation with previously found edge points, and have $m(i,j)$ above a certain threshold. The gray level of the point and its neighbors can be used to vary this threshold to minimize the illumination variance effect.

This approach is similar to Davis' sequential edge detection procedure [1] and Rosenfeld and Kak's omnidirectional tracking algorithm [4]. Davis pointed out that the critical components in detecting edges sequentially are the choice of a good starting point, the dependence structure, and the terminating criterion. Rosenfeld and Kak's omnidirectional tracking algorithm for curve following scans the picture systematically and looks for points that meet a detection criterion. When such a point is found, a tracking criterion is applied to the neighborhood of that point to determine a next point. If no such point can be found, tracking of that branch terminates. The algorithm ends when the systematic scan is completed.

In our decision strategy, the picture is scanned row by row. Our criterion for a good starting point is for the strength value to be above a preset threshold. Two criteria for determining a next point has been considered, they use both the strength value and the edge direction.

We have implemented the decision strategy by a tracing algorithm that is similar to the one described by Graham [5]. Starting at the upper left corner, the algorithm searches for a point with $m(i,j)$ above a preset contour start threshold and which is not already an edge point. Once a contour start point is found, the algorithm begins to trace an edge segment.

Consider a 3 × 3 window centered on an edge point that has just been found. If the edge direction at the center pixel points in the direction d ($d \varepsilon \{0,1,\ldots,7\}$) , only the 3 neighbors corresponding to $(d - 1)$, d, and $(d + 1)$ (mod 8) are considered as candidates for the next point on the edge. Two criteria for determining the next point have been considered: one emphasizing the strength value and the other emphasizing the direction value. We consider both approaches below.

After determining the next edge point, the window is moved to that pixel and another next edge point is searched for. If no next point can be found, the edge segment is terminated.

The newly found contour is merged with any previously found contours if they are connected to each other. If the contour length is below a minimum value, the contour is rejected. The algorithm then returns to the neighbor of the contour start point to search for the next start point. The algorithm terminates when the lower right corner is reached.

3.1.1. Strength Based Criterion

The strength based criterion checks a connectivity condition first. The connectivity condition is satisfied if

$$d - 1 \leq d_k \leq d + 1 \quad (mod \quad 8) \tag{8}$$

where d_k = edge direction of the neighbor under consideration

d = edge direction of center pixel

In this case, the edge direction values have already been quantized to be one of eight values discussed above. Among those pixels that meet this connectivity condition, the next point is chosen to be candidate i if

$$m_i > m_j \quad for \quad all \quad j \neq i$$

$$and \quad m_i > \; preset \; edge \; point \; threshold. \tag{9}$$

where m_i, m_j are strength values of candidates i and j.

3.1.2. Direction Based Criterion

The direction based criterion first rejects those candidates with strength *below* an edge point threshold. The connectivity condition for this criterion is based on previous work by Kitchen and Rosenfeld [6].

$$Let \quad a(\alpha,\beta) = \left| 1 - \frac{|\alpha - \beta|}{\pi} \right| \tag{10}$$

where α,β are angles with values between 0 and 2π

$$\text{then} \quad r_k = a(d, d_k) \times a\left(\frac{\pi}{4}.k, d\right) \tag{11}$$

$$\text{for} \quad k = d,\ d+1,\ d-1 \quad (\text{mod } 8)$$

$$\text{where} \quad d_k = \text{edge direction of neighbor } k$$

$$d = \text{edge direction of the center pixel.}$$

In this case, the edge direction values range from 0 to 2π. The next point is chosen to be candidate i if

$$r_i > r_j \ \text{for all} \ j \neq i$$

$$\text{and} \quad r_i > \text{preset continuity threshold.} \tag{10}$$

3.2. Threshold Selection

Two different criteria for setting the edge point threshold for the strength values in Sections 3.1.1 and 3.1.2 have been considered.

(1) A constant threshold. This is computationally efficient. However, when the illumination is not even throughout the picture, edges in areas with poor lighting tend to be missed.

(2) A locally adaptive threshold [2] is defined for every point (i,j). Consider the lowpass mask:

$$M_0 = \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \tag{13}$$

The locally adaptive threshold (LAT) for pixel (i,j) is defined as

$$LAT(i,j) = \frac{\max_i \left|(W_i, X)\right|}{(M_0, X)} \tag{14}$$

where W_i's are the template masks.

3.3. Edge Detectors Implemented

From the above discussion, different edge detectors can be developed using various combinations of spatial operators, next-point criteria, and threshold selection criteria. Three edge detectors have been implemented:

(1) isotropic gradient masks, the strength based next-point criterion, and fixed threshold.

(2) isotropic gradient masks, the direction based next-point criterion, and fixed threshold.

(3) 3-level template masks, the strength based next-point criterion, and locally adaptive thresholds.

4. Performance

The desirable features of the output of an edge detector include accuracy of edge location, good continuation and thinness of edge segments. The new edge detectors that were presented above were evaluated using Pratt's figure of merit [7] and Kitchen and Rosenfeld's measure [6]. The Sobel operator is also included in the tests for comparison.

White Gaussian noise was added to test images (discussed below) at signal to noise ratios of 100, 50, 20, 10, and 5. Following Pratt's notation [7], the signal to noise ratio SNR is defined as:

$$SNR = \left[\frac{h}{\sigma}\right]^2 \tag{15}$$

where h is the edge height and σ is the standard deviation of the noise.

4.1. Pratt's Figure of Merit

Pratt's figure of merit assesses edge location accuracy by the displacement of detected edge points from an ideal edge. The figure of merit is defined by:

$$R = \frac{1}{I_n} \sum_{i=1}^{I_A} \frac{1}{1+\alpha d^2} \tag{16}$$

where $I_n = \max(I_I, I_A)$

I_I = number of ideal edge points

I_A = number of actual edge points

d = displacement of actual edge points from ideal edge

α = scaling constant.

For our tests, we choose $\alpha = \frac{1}{9}$.

The test image used was 256 × 256 pixels and consisted of a left panel with gray level 115, a right panel with gray level 140, and a single central column of gray level 128. The threshold for each edge detector is adjusted to achieve the maximum figure of merit. Figure 4 contains a plot of the measure against SNR.

4.2. Kitchen and Rosenfeld's Measure

This measure is based on the continuation and thinness of the edges. The measure E is defined by

$$E = \gamma C + (1-\gamma) T \tag{17}$$

where C is the continuation measure,

T is the thinness measure and

γ is the scaling constant.

The continuation measure C is the average of the best continuation at the left and at the right. The best continuation at the left is the maximum value of this function:

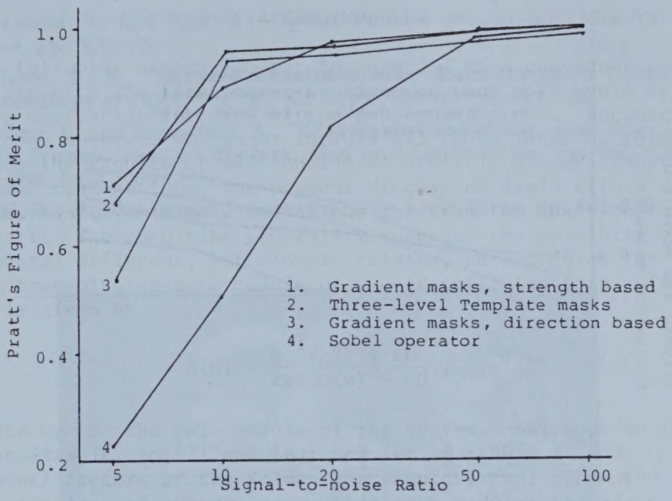


Figure 4. Pratt's Figure of Merit vs Signal-to-noise Ratio

$$L(k) = \begin{cases} a(d,d_k)\times a(\frac{\pi.k}{4}, d + \frac{\pi}{2}) & \textit{if neighbor } k \textit{ is an edge point,} \\ 0 & \textit{otherwise.} \end{cases} \quad (18)$$

where d_k and d are directions returned from the spatial operators for the 3 neighbors to the left of the center pixel relative to the directions returned from the spatial operators, and $a(\alpha,\beta)$ is as defined in Equation 8.

Similarly, the best continuation at the right is the maximum value of

$$R(k) = \begin{cases} a(d,d_k)\times\ a\ (\frac{\pi.k}{4}, d - \frac{\pi}{2}) & \textit{if neighbor } k \textit{ is an edge point} \\ 0 & \textit{otherwise.} \end{cases} \quad (19)$$

where d_k and d are directions returned from the spatial operators for the 3 neighbors to the right of the center pixel relative to the directions returned from the spatial operators.

T is the fraction of the remaining six pixels of the neighborhood that are not edge points.

We chose γ to be 0.8 in our tests. The test image used was a "rings" image similar to the one used by Kitchen and Rosenfeld. This image consists of concentric rings with alternate pixel values of 115 and 140. The image was generated as a 512 × 512 image and then reduced to 256 × 256 pixels by replacing 2 × 2 blocks with the average pixel value of the block. This test image provides edges at all possible orientations. Figure 5 contains a plot of the measure against signal-to-noise ratio.

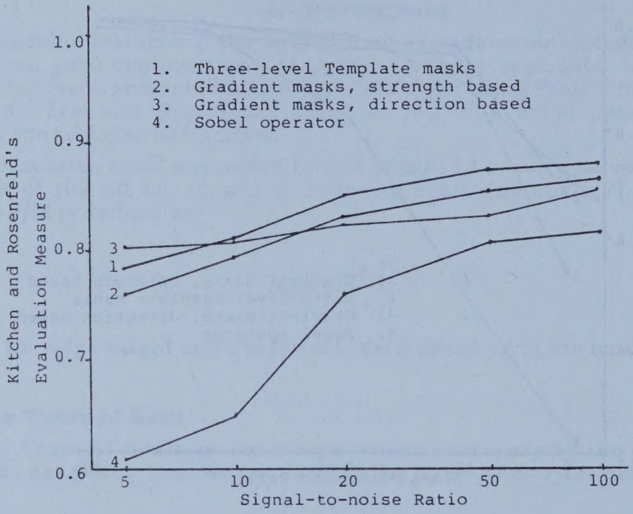


Figure 5. Kitchen and Rosenfeld's Measure vs Signal-to-noise Ratio

5. Discussion

The edge detectors implemented have been applied to real-world images. The results are shown in Figure 6 and compared to the Sobel operator and the Frei-Chen operator. While all operators can detect most of the edge points, the new edge detectors produce thin and connected edges. The results of the tests in Section 4 provides a quantitative measurement of their capabilities to give well-formed edges.

Besides the edge point threshold that had been discussed in Section 3.2, there are several thresholds in our algorithm that can affect the output. The contour start threshold that is used to determine if a contour should be started should be fairly high to assure that a true edge segment is being traced and to avoid generating artifacts. The two threshold system with a much lower edge point threshold used in contour tracing is consistent with our philosophy that the edge direction is primarily responsible for tracing edge segments. Since the connectivity analysis used in contour tracing is a local operation, it is sensitive to noise, hence artifacts can be generated. The minimum contour length value provides some control and can be used to reject isolated points and short contours that frequently are generated by noise.

The continuity threshold in Equation 12 can be adjusted to control the angle that an edge segment can turn. Using some a priori knowledge of the input picture, the various thresholds can be chosen to produce well-formed edges.

6. References

[1] Davis, L., "A Survey of Edge Detection Techniques", *Computer Graphics and Image Processing*, Vol. 4, pp. 248-270.

[2] Robinson, G., "Edge Detection by Compass Gradient Masks", *Computer Graphics and Image Processing*. Vol. 6, pp. 492-501.

[3] Frei, W. and Chen, C., "Fast Boundary Detection: A Generalization and a new Algorithm", *IEEE Transactions on Computers*, pp. 988-998, Oct. 1977.

[4] Rosenfeld, A. and Kak, A., *Digital Picture Processing*, New York:Academic Press, pp. 320-322.

[5] Graham, D. N., "Image Transmissions by Two-Dimension Contour Coding", *Proceedings of IEEE*, Vol. 55, No. 3, pp. 336-347, March 1967.

[6] Kitchen, L. and Rosenfeld, A., "Edge Evaluation Using Local Edge Coherence", *IEEE Transactions on System, Man and Cybernetics*, Vol. SMC-11, No. 9, pp. 597-605, Sept. 1981.

[7] Pratt, W., *Digital Image Processing*, New York: Wiley, pp. 495-699.

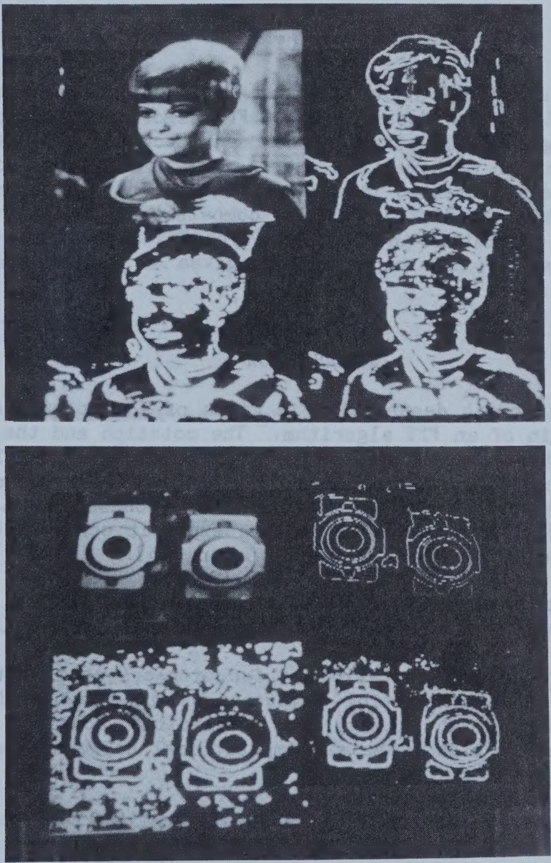


Figure 6. Examples of applying various edge operators.

Upper Left: Original image.

Upper Right: Output from Edge detector (1) in Section 3.3.

Lower Left: Output from Frei-Chen operator.

Lower Right: Output from Sobel operator.

AN ALGEBRAIC DESCRIPTION OF ARRAY IMPLEMENTATIONS OF FFT ALGORITHMS

Lennart Johnsson
Computer Science
California Institute of Technology
Pasadena, California 91125

Danny Cohen
Information Sciences Institute
University of Southern California
Marina Del Rey, California 90291

ABSTRACT

Fast Fourier Transform, FFT, algorithms are interesting for direct hardware implementation in VLSI. The description of FFT algorithms is typically made either in terms of graphs illustrating the dependency between different data elements or in terms of mathematical expressions without any notion of how the computations are implemented in space or time. In this paper a notation that explicitly models the distribution of computations in space and time is used to describe a decimation-in-frequency type FFT algorithm. Expressions in the notation used in this paper can be given an interpretation in the implementation domain.

INTRODUCTION

In this paper an algebraic description is given of an implementation in the space and time domain of an FFT algorithm. The notation and the methodology explicitly model storage, arithmetic, and logic operations. The notation was originaly proposed by Cohen, [Cohen 78], and has later been applied to two-dimensional networks, [Weiser, Davis 80], [Johnsson et. al. 81], [Weiser, Davis 81], and extended to include explicit control [Cohen, Tyree 79], [Johnsson, Cohen 81a], and [Johnsson, Cohen 81b].

A mathematical approach allows for a precise definition, synthesis, analysis, and manipulation of concurrent computational networks.

The traditional mathematical notation is function-oriented, with no regard to issues related to distributing computations in space and time. By introducing the notion of storage and explicitly modeling control of computational devices a wide class of computational networks can be treated formally.

Without using a storage operator it is possible to model only combinational state-free networks composed of only memory-free functional units. However, the use of a storage operator allows for the mathematical modeling of the behavior of any computational network using both combinational logic and storage.

The reason for using such a mathematical notation is the existence of a correspondence between statements in the notation and computational networks. Because of this mapping between the symbolic expressions and the networks, it is possible to use formal symbolic manipulations for the synthesis and the analysis of networks. This correspondence holds up to some trivial transformations.

A major concern in design is the trade-off between time and space. A full instantiation of an algorithm <u>in space</u> is a <u>fully concurrent</u> implementation, whereas a full instantiation <u>in time</u> is a <u>fully sequential</u> implementation.

DEFINITIONS

Following [Cohen 78], the operator Z is defined by:

$$Zx(j) = x(j-1)$$

Let $Z^k = ZZ^{k-1}$

Then $Z^k x(j) = x(j-k)$

Define Z^{-1} to be the inverse of Z such that

$$ZZ^{-1} = Z^{-1}Z = Z^0 = I,$$ where I is the identity operator.

If {x} is a sequence of data elements such that element x(j) precedes x(j+1), then j can be interpreted as time, Z as a <u>delay</u>, and its inverse as a <u>prediction</u>.

The operator Z is a model of a storage cell such as a flip-flop or a shift register cell.

Z^N models a shift register of length N.

The operator Z commutes with time-independent functions:

$$ZF(U) = F(ZU) \qquad (1)$$

For example, if U={u,v} then ZF(u,v) = F(Zu,Zv)

Figure-1 shows a graphical representation of this commutative distributive property.

```
           *****                          *****
           *   *<-- u                     *   *<--[Z]<-- u
W <--[Z]<--* F *        <==>        W <---* F *
           *   *<-- v                     *   *<--[Z]<-- v
           *****                          *****
```

Figure 1: ZF(u,v) = F(Zu,Zv)

If C is a constant, then ZC=C and the following holds:

$$Z(CX) = (ZC)(ZX) = C(ZX) = CZX$$

Hence, as operators Z and C commute.

Multiplexing is defined as a choice between two given arguments according to the value of a given control signal. Let M(X,Y,U) be the multiplexing choice between X and Y according to U, then it is formally modelled by:

$$M(X,Y,U) = \left\{ \begin{array}{l} X, \text{ if } U=1 \\ Y, \text{ if } U=0 \end{array} \right\} = UX + U'Y \tag{2}$$

A graphical representation of the MUX is shown in Figure-2.

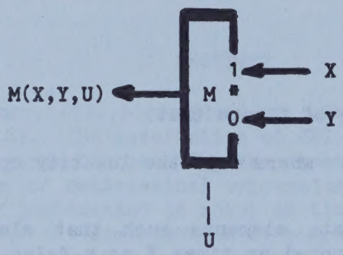


Figure 2: Graphic representation of a multiplexer

The expression UX+U´Y is formally a sum of two products of which at least one is equal to zero. There are many ways to implement this function and the optimal choice depends on the technology used. We do not suggest that UX and U´Y should be implemented by multiplication, by "enabling", or by a logical-AND, nor do we suggest that the combination of these "products" be implemented by addition, logical-OR or exclusive-OR.

When there is a need to emphasize this point we often use the "#"-sign instead of the "+"-sign (e.g., UX#U´Y) to indicate the combination of two entities, of which only one is actually valid (enabled, non-zero, etc).

Throughout this note upper case symbols (like X) denote signals and operators, whereas lower case symbols (such as x) denote specific values.

THE DISCRETE FOURIER TRANSFORM

Throughout this paper we use the following formulation of the Discrete Fourier Transform (DFT):

$$y(m,k) = \sum_{j=0}^{N-1} w^{jm}\, x(k-j) \qquad m=0,1,2...N-1 \tag{3}$$

where y(m,k) is the mth DFT component for a window of length $N=2^n$ ending at the time k. In the context of N-point windows w is defined by:

$$w = w_N = e^{2\pi i/N}$$

Usually the subscript, N, is omitted.

The definition is continuous (i.e., it is valid for any value of k) but we are interested to compute the DFT only for non-overlapping contiguous windows of size N. The set of values x(k-j), j=0,1,2,...,N-1 is assumed to enter sequentially. The implementation network for the decimation-in-frequency (DIF) type FFT that is derived below computes y(m,k), m=0,1,...,N-1 sequentially. The phase of the FFT windows are arbitrarily chosen so that k=N,2N,...

Description of one DIF butterfly

The property of the DFT coefficients that is used extensively in decimation-in-frequency type algorithms is

$$\text{Then} \quad w^{(j+N/2)m} = \begin{cases} w^{jm} & \text{if m is even} \\ -w^{jm} & \text{if m is odd} \end{cases} \tag{4}$$

Use this in (3) and get:

For even m:

$$y(2m,k) = \sum_{j=0}^{N/2-1} w^{2jm} \, [1 + Z^{N/2}] \, x(k-j) \qquad m=0,1,..,N/2-1$$

For odd m:

$$y(2m+1,k) = \sum_{j=0}^{N/2-1} w^{2jm} \, w^{j} \, [1 - Z^{N/2}] \, x(k-j) \qquad m=0,1,..,N/2-1 \tag{5}$$

Define r(k-j) and s(k-j) by:

$$r(k-j) = [1 + Z^{N/2}] \, x(k-j) \qquad j=0,1,...,N/2-1$$

and

$$s(k-j) = w^{j} [1 - Z^{N/2}] \, x(k-j) \qquad j=0,1,...,N/2-1 \tag{6}$$

Equation (6) defines the first set of N/2 butterfly computations according to the radix-2 decimation-in-frequency approach, [Cooley, Tukey 65]. Figure-3 shows an instantiation of one butterfly computation in space. The computations of successive butterflies is spread in time, i.e., the computations for consecutive values of j is performed sequentially.

The two signals R and S are both computed in the second half of the window for the DFT.

We choose to sequentialize R and S to create a single output stream, $X_{N/2}$, by delaying the values of S by N/2 cycles.

The subscript L of a signal G_L means that this signal is treated as a sequence of consecutive windows, each of length L. Hence, the original signal, X, should be considered as X_N.

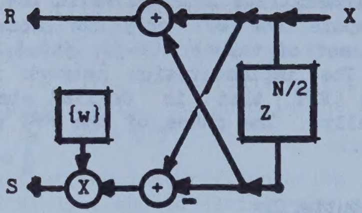


Figure 3: A DIF butterfly, as defined in (6)

Let t=k-j. Then it follows from the definition of R that for $k-N/2<t\le k$

$$x_{N/2}(t) = r(t) = [1 + Z^{N/2}]\, x_N(t) \tag{7}$$

Since S follows R, $x_{N/2}(t)$ for $k<t\le k+N/2$ is defined by

$$x_{N/2}(t) = Z^{N/2}\, s(t) = Z^{N/2}\, [w^{k-t}\, [1 - Z^{N/2}]]\, x_N(t) \tag{8}$$

Define $[t]\equiv t \bmod(N)$ and $\langle t\rangle = t-[t]$. At any time, t, $\langle t\rangle = k$ and $t-k=[t]$.

The two expressions for $X_{N/2}$ can be combined by introducing a multiplexor controled by a periodic signal U_N.

$$u_N(t) = \begin{cases} 1 & 0 \le [t-1] < N/2 \\ 0 & N/2 \le [t-1] < N \end{cases} \tag{9}$$

The two expressions (7) and (8) can now be combined into

$$x_{N/2}(t) = (\, u'_N(t)[1 + Z^{N/2}] \;\#\; u_N(t)\, Z^{N/2}\, [w^{[-t]}\, [1 - Z^{N/2}]])\, x_N(t) \tag{10}$$

It is easily verified that equation (10) is equivalent to (7) and (8).

A network implementing (10) is shown in Figure-4.

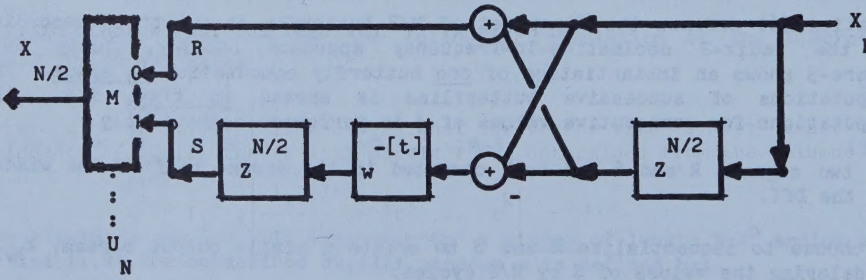


Figure 4: Computing $X_{N/2}$ from X_N

A complete implementation of a DIF algorithm

The even components of y(*,k) are the N/2-point DFT of the R and the odd components are the N/2-point DFT of S. Hence, they can be computed (in this order) by performing N/2-point DFT's on the signal $X_{N/2}$, which is the composition of the R's and the S's.

The same transformation, with N/2 instead of N, computes $X_{N/4}$ which is composed of the new R and S for each N/2-point window of the signal $X_{N/2}$.

By iterating this process logN times all the components of y(*,k) are obtained in reversed-binary order. If B(N) denotes the operator that transforms the signal X_N into the signal $X_{N/2}$, then a complete DFT is performed by $B(2)B(4)...B(N/2)B(N)X_N$.

Each B(N) needs a control function, U_N. Since it is easier to half the frequency of square waves than to double it, U_N is derived from $U_{N/2}$ unlike $X_{N/2}$ which is derived from X_N. Therefore, the control signal propagates in the opposite direction to the propagation of the data, X.

A simplified drawing of Figure-4 is shown in Figure-5, depicting the operator B(N).

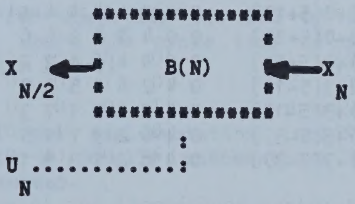


Figure 5: Computing $X_{N/2}$ from X_N

An array for N=8 is shown in Figure-6. The coefficients and the control functions involved in that array are shown in Figure-7.

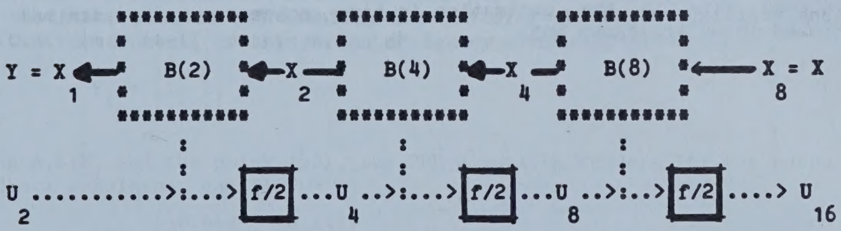


Figure 6: A DIF array for N=8

The notation m(i+j) is a short hand notation for $w_8^m[X(i)+X(j)]$.

t	j	U_8	X_4	j	U_4	X_2	j	U_2	X_1
1	7	0		6	0		4	0	
2	6	0		4	0		0	1	
3	5	0		2	1		4	0	
4	4	0		0	1		0	1	
5	3	1	0(5+1)	6	0		4	0	
6	2	1	0(6+2)	4	0		0	1	
7	1	1	0(7+3)	2	1	0[0(7+3)+0(5+1)]	4	0	
8	0	1	0(8+4)	0	1	0[0(8+4)+0(6+2)]	0	1	0[0(8+4)+0(6+2)]+0[0(7+3)+0(5+1)]
9	7	0	3(5-1)	6	0	2[0(7+3)-0(5+1)]	4	0	0[0(8+4)+0(6+2)]-0[0(7+3)+0(5+1)]
10	6	0	2(6-2)	4	0	0[0(8+4)-0(6+2)]	0	1	0[0(8+4)-0(6+2)]+2[0(7+3)-0(5+1)]
11	5	0	1(7-3)	2	1	0[1(7-3)+3(5-1)]	4	0	0[0(8+4)-0(6+2)]-2[0(7+3)-0(5+1)]
12	4	0	0(8-4)	0	1	0[0(8-4)+2(6-2)]	0	1	0[0(8-4)+2(6-2)]+0[1(7-3)+3(5-1)]
13	3	1		6	0	2[1(7-3)-3(5-3)]	4	0	0[0(8-4)+2(6-2)]-0[1(7-3)+3(5-1)]
14	2	1		4	0	0[0(8-4)-2(6-2)]	0	1	0[0(8-4)-2(6-2)]+2[1(7-3)-3(5-3)]
15	1	1		2	1		4	0	0[0(8-4)-2(6-2)]-2[1(7-3)-3(5-3)]
16	0	1		0	1		0	1	

	8	4	6	2	7	3	5	1	8	7	6	5	4	3	2	1
0[0(8+4)+0(6+2)]+0[0(7+3)+0(5+1)]	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0[0(8+4)+0(6+2)]-0[0(7+3)+0(5+1)]	0	0	0	0	4	4	4	4	0	4	0	4	0	4	0	4
0[0(8+4)-0(6+2)]+2[0(7+3)-0(5+1)]	0	0	4	4	2	2	6	6	0	2	4	6	0	2	4	6
0[0(8+4)-0(6+2)]-2[0(7+3)-0(5+1)]	0	0	4	4	6	6	2	2	0	6	4	2	0	6	4	2
0[0(8-4)+2(6-2)]+0[1(7-3)+3(5-1)]	0	4	2	6	1	5	3	7	0	1	2	3	4	5	6	7
0[0(8-4)+2(6-2)]-0[1(7-3)+3(5-1)]	0	4	2	6	5	1	7	3	0	5	2	7	4	1	6	3
0[0(8-4)-2(6-2)]+2[1(7-3)-3(5-3)]	0	4	6	2	3	7	1	5	0	3	6	1	4	7	2	5
0[0(8-4)-2(6-2)]-2[1(7-3)-3(5-3)]	0	4	6	2	7	3	5	1	0	7	6	5	4	3	2	1

Figure 7: Index computation for the N=8 example

The above algorithm is akin to the decimation-in-frequency type FFT. Algorithms akin to the decimation-in-time scheme can be derived and described in an analogous way.

SUMMARY

This paper develops and proves an implemention of an FFT-algorithm that is partially instantiated in time, partially in space. A decimation-in-frequency algorithm was used as the example, but the notation and methodology applies to other FFT-algorithms as well as entirely different sequential/concurrent algorithms.

The particular implementation described here accepts a sequence of input values, one per cycle, and computes the DFT for consecutive windows of length N. Every N cycles it starts to output a DFT of a new input window, in a binary-reversed order. The first component of a DFT is being output during the cycle when the last data item of the window is being input. Hence, the computations of two successive windows are concurrently being pipelined through the array.

This implementation is based on the use of several similar modules, one per each power of 2 in the window size. For example, 10 such modules are needed for windows of size N=1,024 . The output module computes a DFT of size 2 irrespective of the size of the total transform. The module next to it computes a DFT of size 4, etc. For windows of size N=512 the input module of the 1,024 point array is not needed. That module has as much (shift register) storage as the combined storage of the other 9 modules .

The time required for computing the DFT by this array is N-cycles for windows of size N, where the cycle time is determined by the available complex-multiplication speed.

The development of the implementation including the necessary control signals is done in a totally mathematical and formal fashion. The correctness is guaranteed.

ACKNOWLEDGEMENT

This work was sponsored by the Defense Advanced Project Agency (DARPA) under Contract No. MDA903-81-C-0335 with USC/Information Sciences Institute and Contract N00014-79-C-0597 with the California Institute of Technology. Views and conclusions contained in this paper are the author's and should not be interpreted as representing the official opinion or policy of DARPA, the U.S. Government, or any person or agency connected with them.

REFERENCES

[Cohen 78] Cohen D.
Mathematical Approach to Iterative Computational Networks.
In Proceedings of the Fourth Symposium on Computer Arithmetic, pages 226-238. IEEE Computer Society, October, 1978.

[Cohen, Tyree 79]
Cohen D., Tyree V. C.
VLSI System for Synthetic Aperture Radar (SAR) Processing.
In Proceedings of SPIE, vol. 186, pages 166-177. Society for Photo-Optical Instrumentation Engineers, August, 1979.

[Cooley, Tukey 65]
Cooley J. W., Tukey J. W.
An Algorithm for Machine Calculation of Complex Fourier Series.
Mathematics of Computation 19:297-301, 1965.

[Johnsson et. al. 81]
Johnsson, S. Lennart, Weiser, U., Cohen, D., and Davis, A.
Towards a Formal Treatment of VLSI Arrays.
In Proceedings of the Second Caltech Conference on VLSI, pages . Caltech Computer Science Department, January, 1981.

[Johnsson, Cohen 81a]
Johnsson S. L., Cohen D.
A Mathematical Approach to Modelling the Flow of Data and Control in Computational Networks.
In H. T. Kung, B. Sproull, G. Steele (editor), VLSI Systems and Computations, pages 213-225. Computer Sciences Press, October, 1981.

[Johnsson, Cohen 81b]
Johnsson S. L., Cohen D.
A VLSI Approach to Real-Time Computation Problems.
In Proceedings of SPIE, vol. 298. Society for Photo-Optical Instrumentation Engineers, August, 1981.

[Weiser, Davis 80]
Weiser, Uri and Davis, Alan L.
Mathematical Representation for VLSI Arrays.
Technical Report UUCS-80-111, University of Utah, Computer Science Department, September, 1980.

[Weiser, Davis 81]
Weiser Uri, Davis Al.
A Wavefront Notation Tool for VLSI Array Design.
In H. T. Kung, B. Sproull, G. Steele (editor), VLSI Systems and Computations, pages 226-234. Computer Sciences Press, October, 1981.

ESTIMATION OF MOTION PARAMETERS FROM LOCAL MEASUREMENTS OF VELOCITY COMPONENTS[1]

EDWARD J. DELP and PAUL H. EICHEL
Department of Electrical and Computer Engineering
and
Computer, Information, and Control Engineering Program
University of Michigan
Ann Arbor, Michigan 48109

ABSTRACT

Estimating the motion parameters of moving objects in temporal sequences of images is an important problem in time-varying imagery. Because of its computational simplicity, the method of linear least squares is attractive for the case of two-dimensional motion with scale change, i.e. rigid objects moving in a plane parallel to the image plane plus zooming. Its use is not limited to a spatial gradient based motion detector as widely reported in the literature. Other pixel based motion detectors provide the same local velocity information as do the gradient operators and may similarly be employed in parameter estimation. Each of these motion detectors possesses particular advantages regarding scene segmentation, resolution, and edge detection. In this paper we show that they are mathematically similar in the nature of the information they furnish regarding the local velocity field.

1.0 INTRODUCTION

Numerous motion detectors have been proposed which involve only local computations on a sequence of images [3], [5], [9], [10], [12], [13], [17]. These computations yield incomplete information regarding the local velocity field, providing only scalar information such as the inner product of the velocity with some other local vector quantity. Additional computation is required to infer the macroscopic motion parameters from these microscopic quantities. Newmann [14], Schalkoff [17], and Prazdny [16] have applied a straightforward least squares method to obtain motion parameter estimates directly from the output of one such motion detector based on the spatial gradient. In this paper we show that the same approach may be extended to the outputs of two other detectors: the $\nabla^2 G$ operator and difference pictures.

The use of this motion estimation technique is appropriate where the object(s) to be tracked may be assumed to be of constant shape, i.e. rigid and moving in a plane parallel to the image plane. Scale change or zooming as a result of translation perpendicular to the image plane is also permitted, but rotation about an axis parallel to the image plane presents new views of an object and therefore, in general, results in a change of shape. Motion in this latter sense gives rise to non-linear terms in the equations of motion and necessitates the use of more sophisticated techniques to solve for the parameters [20].

2.0 MOTION DETECTORS

Three pixel-based motion detectors have been prominent in the literature. Least squares motion parameter estimation has only been applied to the spatial gradient method. In this section, we examine the information provided by each of the motion detectors regarding the local velocity field, emphasizing in particular their application to the motion parameter estimation problem.

---

[1] This work was partially supported by the Center for Robotics and Integrated Manufacturing, Robot Systems Division, College of Engineering, University of Michigan and the Air Force Office of Scientific Research under Grant F49620-82-C-0089.

2.1 Spatial Gradient

The spatial gradient is a very popular method for detecting motion. Based on early work by Limb and Murphy [10], Cafforio and Rocca [3], and Netravali and Robbins [13], this method obtains velocity information at a point from the interframe change in the intensity at that point and the local intraframe spatial gradient. The underlying concept is that, to a first order, the change in pixel intensity at a point $\vec{x} = (x,y)$ is equal to the inner product of the local velocity and spatial gradient vectors at that point:

$$\Delta f(\vec{x}) = -\nabla f(\vec{x}) \cdot \vec{V}(\vec{x}) \tag{1}$$

where $\vec{V}(\vec{x})$ is the local velocity at $\vec{x}$, $f(\vec{x})$ is the image intensity function, ∇f is the spatial gradient of the intensity function, and Δf is the change in the intensity from one frame to the next. Knowledge of the gradient as estimated by a gradient operator, such as the Sobel operator, may then be used to infer the component of velocity in the direction of the gradient:

$$\vec{V} \cdot \vec{n} = -\frac{\Delta f(\vec{x})}{|\nabla f|} \tag{2}$$

where $\vec{n}$ is a unit vector in the direction of the gradient. However, an unambiguous specification of both the magnitude and direction of the velocity cannot be accomplished [19].

In the past, authors have either used clustering techniques to solve for the motion parameters [4], or have assumed the velocity field to be constant in small neighborhoods and estimated the x and y components of this local velocity by two-dimensional least squares. Horn and Schunck [5] allow the velocity field to vary slowly over the neighborhood. Tsai and Huang [20] calculate general three-dimensional motion parameters based on these neighborhood estimates of the velocity field. Under the restriction of two-dimensional motion with change of scale, Newmann [14], Schalkoff [17], and Prazdny [16] have applied linear least squares to directly compute the motion parameters from the gradient information without first obtaining neighborhood velocity estimates, as described in Section 3.

2.2 The Laplacian Operator

Marr and Ullman [12] have applied the static edge detection technique of Marr and Hildreth [11] to the task of motion detection. The static edge detector they used is the $\nabla^2 G$ operator, where G represents a Gaussian spatial filter. Since the Laplacian, or second spatial derivative (which is isotropic), undergoes a zero crossing where the local intensity gradient is greatest in magnitude, the contours traced out by the loci of these zero-crossings represent accurate and "thinned" edges in the image. The blurring of the Gaussian filter counteracts the noise enhancement due to the high frequency response of the Laplacian. It also provides some selectivity in the spatial resolving capability of the detector.

Motion of these edges can be detected by finding the time derivative of the image convolved with the $\nabla^2 G$ operator. The time derivative may be approximated by subtracting a pair of convolved images of a temporal sequence. Because of the monotonic nature of the convolution output at the zero-crossing, the sign of the direction of motion is unambiguously specified by the sign of the derivative, i.e. whether the zero-crossing contour is moving to the "left" or to the "right". This information alone has proved sufficient to perform some motion analysis [1].

The magnitude of the time derivative along with knowledge of the slope of the zero-crossing can be used to find, to a first order approximation, the component of velocity normal to the zero-crossing contour much like the case of the spatial gradient technique:

$$\vec{V} \cdot \vec{n} = \frac{\frac{d}{dt}(\nabla^2 G * f)}{m} \tag{3}$$

where $\frac{d}{dt}(\nabla^2 G * f)$ is estimated as indicated above and m is the slope of the convolved image perpendicular to the zero-crossing contour. As before [19], the complete velocity field cannot be specified by this method, a situation referred to by Marr and Ullman as the aperture effect [12].

The important point to note is that the computation of $\frac{d}{dt}(\nabla^2 G * f)$ at the zero-crossing contours results in a scalar quantity: the inner product of the velocity with a local vector quantity. That quantity is the unit normal to the zero-crossing contour whereas in the spatial derivative scheme it was the gradient. Since the direction of the contour can be estimated from local calculations, the information available for least squares estimation of the motion parameters is identical in nature to that obtained by the gradient approach, namely the component of velocity along some known direction.

Although the $\nabla^2 G$ approach involves more computation, its use may be indicated by other requirements. It is an effective edge detector. For this reason, the contours of the operator provide connected and disjoint groups of data points before the least squares process, making it easier to segment multiple objects moving with different parameters. The gradient-based approach provides no such segmentation clues prior to motion processing. Therefore, where multiple objects must be tracked, the image must be divided up into small regions to be processed individually [6].

2.3 Difference Pictures

Difference pictures provide a relatively simple means of locating areas of an image undergoing change at the pixel level. They have been employed by several authors in the context of dynamic scene analysis especially for segmentation [8],[9]. We show that they may also be used as motion detectors for the least squares motion estimation technique.

Difference pictures in their simplest form may be generated by subtracting one frame of a sequence from a successor, and thresholding the result. Thus, the difference image is binary, with 1's at those pixel locations where the intensity change is larger than the threshold and 0's elsewhere. If the sequence depicts a moving object of sufficiently high contrast with its surroundings, the 1's will fall in those regions that were either newly covered or newly uncovered by the object in the interframe interval. These regions will be referred to as difference regions.

Because of thresholding, difference regions may be generated along interior edges of high contrast as well as the exterior boundaries of the object. In any case, the thickness of a difference region associated with a high contrast boundary is related to the motion perpendicular to the boundary at that point. Motion along the boundary does not contribute to the difference region. Once again we are given information on the component of velocity normal to an edge segment and can estimate the direction of that edge segment from the orientation of surrounding elements. As before, this information may be processed by the method of least squares to yield the motion parameters.

An ambiguity exists in the simple difference pictures described above regarding the sign of the displacement. This may be remedied by treating the newly covered and uncovered regions differently, thus preserving the sense of the motion. Two thresholds, symmetrical about zero, are employed instead of one.

As in the case of the $\frac{d}{dt}\nabla^2 G$ operator, difference pictures yield valuable information regarding object segmentation before the motion parameter estimation process. The difference regions may be readily tested for connectiveness, and the separate regions of the image individually processed.

Unlike both of the motion detectors discussed above, difference pictures can be reliable even where the interframe displacements are relatively large. Since the gradient-based and Laplacian-based techniques retain only the first-order terms of their respective Taylor Series expansions, the displacements resulting from these computations are reliable only for small values. This is especially true near prominent features, such as edges, in the image where the spatial gradient is large. Difference pictures can yield reliable displacement estimates of many pixels depending on object size and shape. The resolution of the difference regions suffers from the inherent spatial quantization of the image; sub-pixel displacements may not be detected at all.

3.0 COMPUTING MOTION ESTIMATES

In this section we develop the equations of motion. It will be seen that they are linear in the motion parameters, thus facilitating a least squares method of solution. The reader is referred to [18] for an alternative development based on the affine transform or to [7] for another approach.

When a rigid object is constrained to move in a plane, the velocity, $\vec{V}(\vec{x})$, at any point, $\vec{x} = (x,y)$, of the object may be decomposed into a translational component, $\vec{V}_t$, and an angular component, $\vec{\omega}$, such that:

$$\vec{V}(\vec{x}) = \vec{V}_t + \vec{\omega} \times \vec{r} \tag{4}$$

where $\vec{\omega}$ is the axis of rotation (perpendicular to the image plane), $\vec{r}$ is a vector from the intersection of this axis with the image plane to the point $\vec{x}$, and "×" denotes a vector cross product [2]. Without loss of generality, we may take the axis of rotation to pass through the origin, so that $\vec{r}$ is now identically the positional vector $\vec{x}$.

Zooming, or change of scale, is a result of translation along the coordinate axis perpendicular to the image plane. Depending on sign, such motion contributes a radial velocity component at each point of the object that is proportional to both the scale change coefficient, α, and the distance of the point from the origin, $|\vec{x}|$. This radial component is directed toward the origin for motion away from the camera and vice versa:

$$\vec{V}_r = \alpha\vec{x} \tag{5}$$

Here, positive α indicates a positive scale change or motion toward the camera. Thus, the net velocity at a point in the image plane is given by:

$$\vec{V}(\vec{x}) = \vec{V}_t + \vec{\omega} \times \vec{x} + \alpha\vec{x} \tag{6}$$

As shown in the previous section, the output of the motion detector is actually the inner product of the velocity field with some other local vector quantity whose direction and magnitude are known. Letting $\vec{n}$ denote a unit vector in this known direction, the component of velocity in the direction $\vec{n}$ at a point is:

$$\vec{V}\cdot\vec{n} = \vec{V}_t\cdot\vec{n} + \vec{\omega}\times\vec{x}\cdot\vec{n} + \alpha\vec{x}\cdot\vec{n} \qquad (7)$$

Expanding this equation, we see that:

$$d = (n_x V_{t_x} + n_y V_{t_y}) + (n_y x - n_x y)\omega + (n_x x + n_y y)\alpha \qquad (8)$$

Here, $d = \vec{V}\cdot\vec{n}$ is the detector output, $\vec{n} = (n_x, n_y, 0)$ the direction of the local vector quantity obtained from the motion detector, $\vec{\omega} = (0,0,\omega)$, $V_t = (V_{t_x}, V_{t_y}, 0)$, and $\vec{x} = (x,y,0)$. When the motion detector is based on the gradient, $n_x = \frac{1}{|\nabla f|}\frac{\partial f}{\partial x}$ and $n_y = \frac{1}{|\nabla f|}\frac{\partial f}{\partial y}$, where $|\nabla f| = \left[\frac{\partial f}{\partial x}\right]^2 + \left[\frac{\partial f}{\partial y}\right]^2$; for the $\frac{d}{dt}\nabla^2 G$ operator, $\vec{n}$ is a unit vector normal to the zero-crossing contour; and for the difference pictures, $\vec{n}$ is normal to the difference region boundaries.

We observe that Equation (8) is linear in the four unknowns $[V_{t_x}, V_{t_y}, \omega, \alpha]$ and may be readily solved when the detector output is known for four or more image points. In the over-determined case, we form the set of equations:

$$A\,\bar{x} = b \qquad (9)$$

and solve for the vector of estimated motion parameters $\bar{x} = [V_{t_x}, V_{t_y}, \omega, \alpha]^T$ through the use of the pseudo-inverse of A:

$$\bar{x} = (A^T A)^{-1} A^T\, b \qquad (10)$$

The column vector b is composed of the motion detector outputs, d, and the elements of A are assembled from the estimates of $\vec{n}$ according to Equation (8). Computationally, more accurate numerical answers can be achieved by using the Moore-Penrose generalized inverse of A rather than Equation (10). This approach is furthermore necessary where $A^T A$ is singular [15]. As discussed in [20], the actual motion parameters of the object itself may only be determined to within a common scale factor (except for the rotation parameter).

4.0 AN EXAMPLE

The use of least squares motion estimation using a gradient-based motion detector has been cited in the literature [14],[17], [16]. In this section, we examine its use with a difference picture motion detector. For situations involving relatively high contrast targets, this scheme exhibits very fast results. It is especially attractive for real-time tracking of multiple targets since difference pictures may be generated at video frame rates and segmentation of difference regions prior to motion parameter estimation can also be accomplished very quickly.

As outlined in Section 2.3, the difference picture must furnish: (1) the component of velocity normal to the boundary of a difference region; (2) the angular orientation of that boundary; and (3) the sign of the motion, i.e. whether the object has moved into or out of the region. The orientation of the boundary is quantized to one of eight angles in this simple example. The two neighboring boundary points of a given point are examined and if the three are found to lie on a straight line, the boundary is judged to be locally of that angle ($\frac{\pi}{4}$ i ,i = 0,1,...,7). The "thickness" of the difference region is then measured by counting pixels in a direction perpendicular to the indicated boundary orientation. This yields the displacement normal to the boundary to the resolution of the spatial quantization of the image. Whether the difference region is one that was newly covered or uncovered by the object affects the boundary orientation angle by π.

This collection of boundary point data may be segmented by the natural disjointedness of the difference regions. The data from each difference region is processed individually to estimate its motion parameters. Regions with similar parameter estimates can then be grouped together. This procedure has the advantage that since each individual difference region is likely to be associated with only one physical object, the least squares error, i.e. the Euclidean distance from the observed data to the least squares projection of that data onto the column space of A:

$$e = || A \bar{x} - b || \tag{11}$$

is much smaller than it would be if data from different objects were lumped together. This is because all points on a rigid object move with identical motion parameters. Small projection errors indicate an increased confidence in the parameter estimates obtained by the least squares method.

Figures 1a and 1b are taken from a sequence of temporally spaced frames in which the two prominent objects have been synthetically translated and rotated in a plane perpendicular to the camera axis and translated toward the camera. The exact motion parameters are therefore known. The difference picture generated by these two frames is displayed in Figure 1c. The two shades of the difference regions indicate whether they have been uncovered (whiter) or covered (darker) by a moving object. Note the significant amount of internal detail present in the thresholded image.

The motion parameter estimates generated by the least squares technique using these difference regions as input fall within the one pixel quantization error of the known values. This is illustrated in Figure 1d. In this image, the objects of the first frame have been moved according to the estimated parameters generated by the least squares process. This is then subtracted from the actual second frame to generate a difference picture indicating the discrepancy between the estimated and actual motion of the objects. The thin lines in this picture represent difference regions one pixel in width. The method is very tolerant of the difference threshold chosen and the amount of interframe displacement.

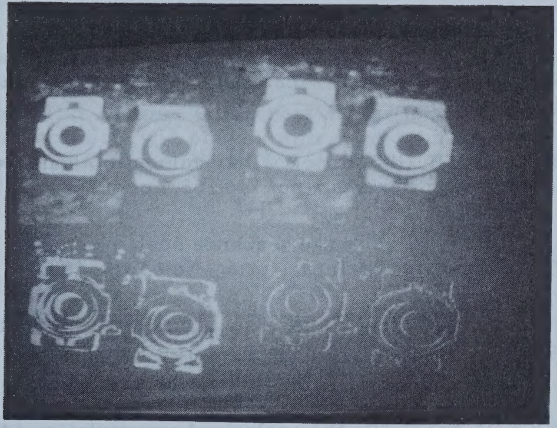


Figure 1: (a) Upper Left: First frame of the sequence. (b) Upper Right: Second frame of the sequence. (c) Lower Left: Difference image between above two frames. (d) Lower Right: Difference image between actual motion and estimated motion.

5.0 REFERENCES

[1] Batali, J. and S. Ullman, "Motion detection and analysis," *Proc. DARPA Image Understanding Workshop*, L.S. Bauman, Ed., Science Application Inc., Nov. 1979, pp. 69-75.

[2] Becker, R. A., *Introduction to Theoretical Mechanics*, McGraw-Hill, 1954.

[3] Cafforio, C. and F. Rocca, "Methods for measuring small displacements of television images," *IEEE Trans. on Info. Theory*, vol. IT-22, pp. 573-579, Sept. 1976.

[4] Fennema, C. L. and W. B. Thompson, "Velocity determination in scenes containing several moving objects," *Computer Graphics and Image Proc.*, vol 9, pp. 301-315, Apr. 1979.

[5] Horn, B. K. P. and B. G. Schunck, "Determining optical flow," *Proc. DARPA Image Understanding Workshop*, pp. 144-156, Apr. 1981.

[6] Huang, T. S. and Y.P. Hsu, "Image sequence enchancement," *Image Sequence Analysis*, T. S. Huang ed., Springer Verlag, New York, 1981.

[7] Huang, T. S. and R. Y. Tsai, "Image sequence analysis; motion estimation," *Image Sequence Analysis*, T. S. Huang ed., Springer Verlag, New York, 1981, pp. 1-18.

[8] Jain, R., and H. H. Nagel, "On the analysis of accumulative difference pictures from image sequences of real world scenes," *IEEE Trans. PAMI*, vol. PAMI-1, pp. 206-214, Apr. 1979.

[9] Jain, R., W. N. Martin, and J. K Aggarwal, "Segmentation through the detection of changes due to motion," *Computer Graphics and Image Proc.*, vol. 11, pp. 13-34, Sept. 1979.

[10] Limb, J. O., and J. A. Murphy, "Estimating the velocity of moving images in television signals," *Computer Graphics and Image Proc.*, vol. 4, pp. 311-327, Dec. 1975.

[11] Marr, D. and E. Hildreth, "Theory of edge detection, *MIT AI-Memo No.518*, Apr. 1979.

[12] Marr, D. and S. Ullman, "Directional selectivity and its use in early visual processing," *MIT AI-Memo 524*, June 1979.

[13] Netravali, A. N. and J. D. Robbins, "Motion compensated television coding: part 1," *Bell Sys. Tech. Journal*, vol. 58, pp. 631-670, March 1979.

[14] Newman, T., "Video target tracking by lie algebra techniques, *Proc. Workshop on Automatic Missile Tracking*, Redstone Arsenal, Alabama, Nov. 1979.

[15] Noble, B. and J. Daniel, *Applied Linear Algebra*, Prentice-Hall Inc., 1977.

[16] Prazdny, K., "Computing motions of planar surfaces from spatio-temporal changes in image brightness," *Proc. IEEE Conf. Pattern Recog. and Image Processing*, June 14-17, 1982, Las Vegas, Nevada, pp. 256-258.

[17] Schalkoff, R. J., "Algorithms for a real-time automatic video tracking system," PhD dissertation, Univ. of Virginia, Charlottesville, Va., May 1979.

[18] Schalkoff, R. J., and E. S. McVey, "A model and tracking algorithm for a class of video targets," *IEEE Trans. PAMI*, vol. PAMI-4, pp. 2-10, Jan. 1982.

[19] Thompson, W. B. and S. T. Barnard, "Lower-level estimation and interpretation of visual motion," *Computer*, vol. 14, pp. 20-28. Aug. 1981.

[20] Tsai, R. Y. and T. S. Huang, "Uniqueness and estimation of three-dimensional motion parameters of rigid objects with curved surfaces," *Proc. IEEE Conf. Pattern Recog. and Image Processing*, June 14-17, 1982, Las Vegas, Nevada, pp. 112-118.

FREQUENCY DOMAIN ALIGNMENT OF DISCRETE-TIME SIGNALS

M. SIMAAN

Interactive Computing Laboratory
Department of Electrical Engineering
University of Pittsburgh
Pittsburgh, PA 15261

ABSTRACT

Alignment of time shifted discrete signals in the time domain using cross correlation techniques without interpolation can only be done to within an integer multiple of the sampling interval. In many applications the accuracy of such techniques is not sufficient. In this paper, we discuss an alignment procedure for discrete signals in the frequency domain which essentially overcomes this difficulty. Two examples are also presented.

INTRODUCTION

Many signal measurement schemes in engineering application consist of sensors which measure a propagating waveform , digitize it and store the data for further processing. Such systems are very common in biomedical, nondestructive evaluation, radar arrays, and seismic applications. Careful alignment of such discrete signals is often required for calculation of the propagation velocity and for estimation of certain ensemble averages.

Alignment in the time domain using cross-correlation technique without interpolation can only be done to within an integer multiple of the sampling interval. This may not provide sufficient accuracy in propagation velocity calculations and may result in loss of high frequency in case of ensemble averaging. Furthermore, the maximum of the cross correlation function may not be easily identifiable. In order to alleviate these problems we discuss in this paper an alignment procedure in the frequency domain. This procedure is related to sinc-function interpolation in the time domain.

PROBLEM FORMULATION

Consider a set of N waveforms defined over $0 \leq t \leq T$, of the form:

$$x_n(t) = s(t-\alpha_n) + \eta_n(t), \; i = 1,\ldots N \qquad (1)$$

Where $s(t)$ and $\eta_n(t)$ for $n=1,\ldots N$ are frequency band limited data and noise respectively and the α_n's are time shift constants. Let $x_n(m)$ for $n = 1,\ldots N$ and $m = 1,\ldots M$ be the corresponding sampled waveforms and let

$T_s = T/(M-1)$ be the sampling interval; chosen so that no aliasing in the frequency domain occurs. Clearly, in general, the time shifts α_n are not integer multiples of T_s and hence, if alignment of the waveforms $x_n(m)$ is to be done in the time domain, the values of α_n can only be recovered up to an integer multiple of T_s. This is due to the fact that, even in the absence of noise ($\eta_n(t) = 0$), the discrete signals $x_n(m)$ will not be identical in shape. For this reason, a definition of what is meant by alignment is needed.

Let $X_n(k)$, $m = 1,\ldots N$ and $k = 1,\ldots M$ be the discrete Fourier transforms [1] of $x_n(m)$ and for a given set of numbers $\tau_1,\ldots \tau_N$, let:

$$e^{j\frac{2\pi k}{M}\tau_n} \qquad \text{for } n = 1,\ldots N \text{ and } k = 1,\ldots M, \tag{2}$$

be the discrete Fourier transforms of frequency domain operators to be applied to $X_n(k)$. We now define alignment as follows:

Definition: If there exists a set of numbers $\tau_1,\ldots\tau_N$, which will minimize the expression:

$$E = \sum_{k=1}^{M}\sum_{n=1}^{N}\sum_{i=1}^{N} \left| X_n(k)e^{j\frac{2\pi k}{M}\tau_n} - X_i(k)e^{j\frac{2\pi k}{M}\tau_i} \right|^2 , \tag{3}$$

then $\tau_1,\ldots\tau_N$ are defined as the alignment time constants, and the discrete waveforms:

$$\hat{x}_n(m) = F^{-1}(X_n(k)e^{j\frac{2\pi k}{M}\tau_n}) \qquad \text{for } n=1,\ldots.N \tag{4}$$

are said to be aligned based on (3).

At this stage note the following:

i) The quantity E is the total energy difference between all possible time shifted pairs of waveforms.

ii) The minimization of E can only be done to within an additive constant to all τ_n's. In order to avoid this, a constraint $\tau_{n_o} = 0$ will be imposed resulting in alignment with respect to the n_oth waveform in the set.

iii) Since τ_n's are not, in general, integer multiples of T_s, the shifted wave forms must be computed based on (4), which essentially is equivalent to interpolation, time-shifting and resampling in the time domain.

SOLUTION PROCEDURE

In order to minimize the energy expression E in (3) with respect to the unknown variables $\tau_1,\ldots\tau_N$, we first note the following identity for complex numbers:

$$\sum_{n=1}^{N}\sum_{i=1}^{N} |z_n - z_i|^2 = 2N\sum_{n=1}^{N} |z_n|^2 - 2\sum_{n=1}^{N}\sum_{i=1}^{N} z_n z_i^{*}. \tag{5}$$

If we let $z_n = X_n(k)\, e^{j\frac{2\pi k}{M}\tau_n}$, we can then write:

$$E = \sum_{k=1}^{M}\{\, 2N\sum_{n=1}^{N} |X_n(k)|^2 - 2\sum_{n=1}^{N}\sum_{i=1}^{N} X_n(k)X_i^{*}(k)\, e^{j\frac{2\pi k(\tau_n-\tau_i)}{M}}\,\} \tag{6}$$

and

$$\frac{\partial E}{\partial \tau_i} = -\frac{4\pi}{M}\sum_{k=1}^{M} \mathrm{Re}\{(\sum_{n=1}^{N} X_n^{*}(k)\, e^{-j\frac{2\pi k}{M}\tau_n})\; jkX_i(k)\, e^{j\frac{2\pi k}{M}\tau_i}\} \tag{7}$$

If now, we define

$$X_i(k)\, e^{j\frac{2\pi k}{M}\tau_i} = A_i(k) + B_i(k) \tag{8}$$

and

$$A(k) = \sum_{n=1}^{N} A_n(k), \quad B(k) = \sum_{n=1}^{N} B_n(k) \tag{9}$$

then (7) reduces to;

$$\frac{\partial E}{\partial \tau_i} = \frac{4\pi}{M}\sum_{k=1}^{M} (A(k)B_i(k) - A_i(k)B(k))k, \quad \text{for } i=1,\ldots N \tag{10}$$

Thus, in order to determine the alignment times $\tau_1,\ldots\tau_N$, the nonlinear equations $\frac{\partial E}{\partial \tau_i} = 0$ for $i = 1,\ldots N$ must be solved. This, in general, cannot be done analytically. Iterative numerical algorithms, however can be used to achieve this. In particular, since the gradient vector $\partial E/\partial \tau$ can be computed analytically, we mention the conjugate gradient algorithm [2], which is a recursive algorithm used extensively in computer minimization of functions of several variables.

COMPUTER IMPLEMENTATION AND EXAMPLES

In order to implement the conjugate gradient algorithm, the algorithm for computing the gradient at the p^{th} iteration at the point $\{\tau_1(p), \tau_2(p),\ldots, \tau_n(p)\}$ would be as follows:

i) determine $A_i^{(p)}(k) = \mathrm{Re}\{X_i(k)\, e^{j\frac{2\pi k}{M}\tau_i^{(p)}}\}$

$$\text{and } B_i^{(p)}(k)=\text{Im}\{X_i(k)e^{j\frac{2\pi k}{M}\tau_i^{(p)}}\}$$

based on (8).

ii) determine $A^{(p)}(k)$ and $B^{(p)}(k)$ based on (9) and,

iii) determine $\left.\frac{\partial E}{\partial \tau_i}\right|_{(p)}$ for i=1,...N

based on (10). Note that the summation in (10), can be replaced by $k=k_L$ to $k=k_H$ where k_L and k_H correspond to the low and high frequencies f_L and f_H over which the energy minimization is to take place. These frequencies are determined as $f_L = \frac{k_L}{T_s}$ and $f_H = \frac{k_H}{T_s}$. In general, the choice of the frequency range f_L, f_H over which alignment is performed, should be the largest frequency interval over which the amplitude spectra of all signals are as similar as possible.

In order to illustrate the effectiveness of this method, we present the following two examples:

Example 1: Consider the waveforms:

$$x_n(t) = e^{-0.1(t-\alpha_n)}\sin 0.2\pi(t-\alpha_n)\ u(t-\alpha_n) + \eta_n(t)$$

for n=1,...N; where t is in msec., α_1, α_2,...α_N are time shifts in msec. denoting the beginning of the decaying sinusoidal signals and $\eta_n(t)$ are zero mean white noise signals. These waveforms are sampled at 1 msec. sampling interval and a total of 128 samples are used for each waveform. A total of six such waveforms have been generated corresponding to values of α_n equal to: $\alpha_1 = 41.5$, $\alpha_2 = 42.2$, $\alpha_3 = 42.8$, $\alpha_4 = 43.4$, $\alpha_5 = 43.5$, and $\alpha_6 = 44.1$ msec. These waveforms are then aligned with respect to the first waveform, using the procedure discussed in the paper. Table (1) shows the results for different values of the signal to noise ratio. In this table τ_{i1} represents the time shifts in msec. between the i^{th} waveform and the first waveform. These values are clearly still within an integer multiple of the sampling rate even when the signal to noise is 0 dB. The maximum error when compared with the exact values of τ_{j1}, are also shown in the table.

	Exact	S/N =	S/N = 20dB	S/N = 10dB	S/N =0 dB
τ_{21}	0.7	0.7119	0.7177	0.7333	0.8161
τ_{31}	1.3	1.3122	1.3187	1.3378	1.4444
τ_{41}	1.9	1.9013	1.9222	1.9666	2.1131
τ_{51}	2.0	2.0001	2.0331	2.1060	2.3500
τ_{61}	2.6	2.6211	2.6453	2.7011	2.9240
Max. Error		1.7%	2.5%	5.3%	17.5%

Table 1.

Example 2. Figure 1, shows a set of 24 waveforms of real seismic data propagating in the earth. The sampling interval is 1 msec. and each waveform contains 0.4 sec. worth of data. Figure 2 shows the data after alignment in the frequency domain. A range of frequency from 4Hz to 100Hz was used in the energy term and the conjugate gradient method converged in 4 iterations.

ACKOWLEDGEMENT

This research was supported in part by a Grant from Gulf Research and Development Company.

REFERENCES

[1] Oppenheim A.V. and Shafer R.W., "Digital Signal Processing," Prentice-Hall Inc., N.J. 1975.

[2] Hestenes M.R. And Stiefel E., "Methods of Conjugate Gradients for Solving Linear Systems" Journal of Research of the National Bureau of Standards, Vol. 49, No. 6, December 1952

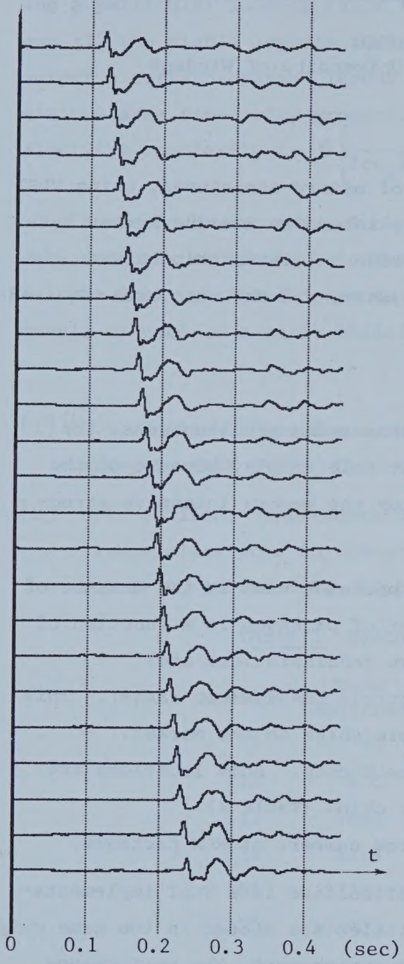


Figure 1: Before Alignment

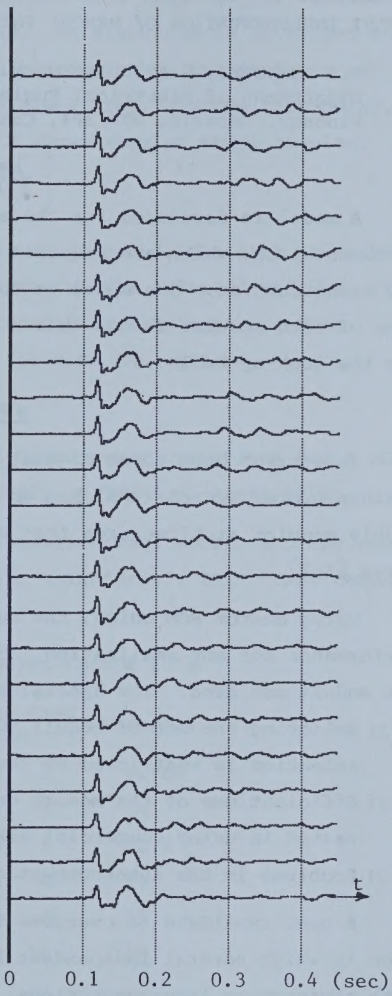


Figure 2: After Alignment

A VLSI IMPLEMENTATION OF MEMORY INTENSIVE RESIDUE NUMBER SYSTEM ARCHITECTURE

M.A. BAYOUMI, G.A. JULLIEN, M.A. SID-AHMED
Department of Electrical Engineering, University of Windsor
Windsor, Ontario, N9B 3P4, Canada

ABSTRACT

A model is developed for the analysis of memory structures, using VLSI implementation, which are required in memory intensive Residue Number System architectures. The model is used to derive a relationship between the size of each modulus in the chosen number system, and the chip area required for the look-up table.

SUMMARY

In current high speed digital signal processing architectures, the residue number system (RNS) has an important role to play because of the highly modular architecrures that evolve from the memory intensive structures [1-2].

Using custom ROM chips, the number of packages used is the measure of performance for any realization. The number of packages is a function of the moduli set used. The special RNS design considerations are:

1) Selecting the set of moduli (which controls the dynamic range). This selection is restricted by the available chips in the market.
2) Efficient use of the memory cells in each chip. Some locations are wasted in using commercial size memory chips (Table 1).
3) Problems in the interconnection of large numbers of ROM packages.

A good candidate to overcome those difficulties is a VLSI implementation in which several independent look-up tables are stored in the same chip with appropriate interconnections. In this environment, the performance measures are the chip area and the time delay imposed by the circuit, i.e., both space and time are included in the performance criterion. In such an environment, connection paths may use substantial amounts of chip area and may contribute a non-negligible time delay.

A model for the memory area and access time is developed based on the hierarchical structures. This model is flexible in that the word width is independent of the capacity; also, the two dimensions are independent and are not restricted. The word width $\log_2(m-1)$, where m is the modulus used, and the number of memory bits required for a look-up table is $m^2 \times \log_2(m-1)$; as an example, using a modulus of 7 the word width is 3 bits and a ROM of 3×49 bits required for look-up table implementation. A mathematical relationship between each RNS modulus and the area required for implementing a particular look-up table is derived. An algorithm is developed to get all the viable layouts for a certain modulus, then the designer can select a certain optimal layout having minimum area, minimum time-delay, or minimum area time-delay product. Table 2 shows some of these results. The algorithm complexity is $O\left(\frac{m^2}{\log_2(\log_2(m-1))}\right)$.

A module having three look-up tables is proposed, the area and time delay of this module is calculated depending on the selected modulus, Fig. 1. The look-up tables contents are filled during fabrication. This module is used as a basic building unit for some RNS systems.

REFERENCES

[1] G.A. Jullien, "Residue number scaling and other operations using ROM arrays", IEEE Trans. Comput., Vol.C-27, pp.325-336, April 1978.

[2] G.A. Jullien, "Implementation of multiplication, modulo a prime number, with applications to number theorteic transforms", IEEE Trans.Comput., Vol.C-29, pp.899-905, October 1980.

TABLE 1 Wasted Bits Using Custom IO's

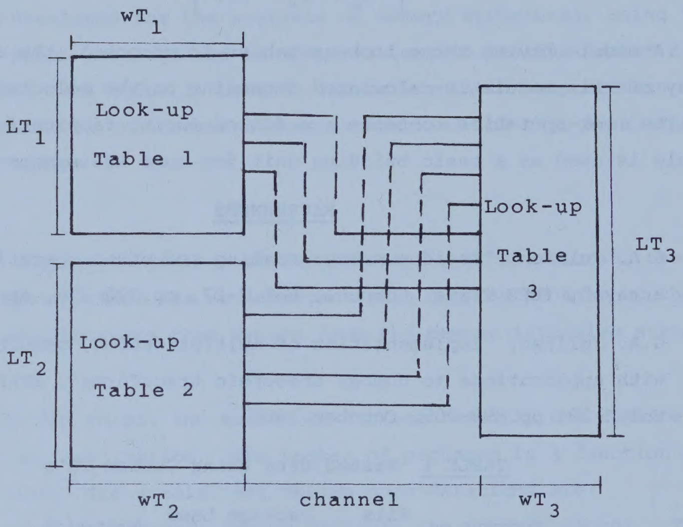
Modulus	Bits Required	Package Used Count	Wasted %
5	75	256*4, 1	92.68
13	676	256*4, 1	34
17	1445	1k * 8, 1	82.4
37	8214	2k * 4, 4	75
61	22326	2k * 4, 4	32
123	105903	1k * 8, 16	19

TABLE 2 The Optimal Layouts of Some Moduli

Modulus	Number of Levels	Length	Width	Area	Time Delay	The Product
7	1	20.49	24.949	511.3	15	7669.5
	1	23.11	22.21	513.41	13	6674.33
	3	64	74.25	4752	9	42768
21	1	62.4	58.4	3646	49	178654
		141.75	296.88	42082	15	631231

Notes: The dimensions of the above table depends on the used technology they are as follows:

Length, width	in minimum design length
Area	in minimum design area
Time	in minimum unit time


wT_1
LT_1
Look-up Table 1
LT_2
Look-up Table 2
Look-up Table 3
LT_3
wT_2
channel
wT_3

$$\text{Length} = \text{larger}\left\{\left(LT_1 + LT_2 + 1\right),\ LT_3\right\}$$

$$\text{Width} = wT_1 + [1 + \log_2(m - 1) + \log_2(m_2 - 1)] + wT_3$$

Fig. 1 A Multi-Look-up Tables Module

INTERNAL ZEROS AND THE SYSTEM MATRIX*

BOSTWICK F. WYMAN	MICHAEL K. SAIN
Department of Mathematics	Department of Electrical Engineering
The Ohio State University	University of Notre Dame
Columbus, Ohio 43210	Notre Dame, Indiana 46556

ABSTRACT

The multivariable transmission zeros of a transfer function were placed in an abstract, module-theoretic context by Wyman and Sain [1] in 1981. This paper seeks an extension of these ideas to the internal zeros of a system. It is observed that the Rosenbrock system matrix defines two distinct, but closely related, zero modules. These modules are examined in regard to their relationships with the transmission-zero module, the input-decoupling zero module, and the output-decoupling zero module.

INTRODUCTION

In this presentation, we consider the algebraic theory of zeros of a linear multivariable system. We work over a field k of scalars, as for example the real number field. Finite-dimensional vector spaces U, X, and Y over k are assumed to embody the controls, the states, and the outputs, respectively. For dynamical descriptions, we build up from the field k a principal ideal domain k[s] of polynomials with coefficients in k and with indeterminate s. Then, from k[s], we construct in the usual way the quotient field k(s). Of course, k(s) is the abstract counterpart of the "field of rational functions having coefficients in k", a staple of classical engineering system theory.

To develop an abstract vector of such rational functions, we can make use of the tensor product. For example, in the case of the control space U, it is possible to write

$$U(s) = U \otimes_k k(s) \ .$$

Here, both U and k(s) are regarded as vector spaces over k, a fact signified by the subscript k on the tensor product symbol ⊗. So defined, U(s) is a vector space over k. It can, however, easily be upgraded to a vector space over k(s), by a mechanism which interprets scalar multiplication in terms of ordinary multiplication in k(s). As a k(s)-vector space, U(s) has finite dimension; as a k-vector space, it does not. This last fact is a consequence of k(s) itself not having finite dimension over k. In a similar way, we can develop a module of abstract vectors of polynomial functions. Again, for the control space, one writes the k-vector space

$$\Omega U = U \otimes_k k[s] \ ,$$

which can be upgraded to a k[s]-module. In the module sense, any k-basis of U can serve as a k[s]-basis of ΩU. Accordingly, ΩU is a free module.

Now consider morphisms $\alpha : V_1 \to V_2$ and $\beta : V_2 \to V_3$ of R-modules

---

*This work was supported by the Ames Research Center of the National Aeronautics and Space Administration, under Grants NAG 2-34 and NAG 2-35.

V_i, $i = 1,2,3$. A sequence of two morphisms may be taken to mean their composition, when it exists, as for example

$$V_1 \xrightarrow{\alpha} V_2 \xrightarrow{\beta} V_3$$

in the case of α and β. Such a sequence is said to be exact when the image of α is equal to the kernel of β. As an illustration, we have the exact sequence

$$0 \longrightarrow \Omega U \longrightarrow U(s)$$

of k[s]-modules, which is another way of saying that ΩU is a k[s]-submodule of $U(s)$. We make further use of exact sequences in later statements of the results of this paper. Of course, similar developments hold for Y, $Y(s)$, and ΩY. Special interest attaches to the quotient

$$\Gamma Y = Y(s)/\Omega Y ,$$

which is also a k[s]-module.

This language, and the mathematical point of view, are rooted in "standard" commutative algebra: modules, morphisms, and exact sequences. For an engineer's-eye introduction to such ideas, the reader may wish to consult [2] or [3].

SYSTEM DESCRIPTION

To describe the system, we wish to make use of the system matrix introduced by Rosenbrock [4]. This system matrix combines the crucial data of a linear multivariable system into an extraordinarily convenient and useful package. Among other successes, the system matrix allows multivariable zeros to be studied numerically as points where an appropriate rank drops; see, for example, [5] and [6]. In the light of these accomplishments, one might hope for significant module-theoretic insights to be built upon the system matrix. However, in large measure, the system matrix remains today just that, a matrix, with much more work needed to exploit its module-theoretic possibilities. One abstract view of the system matrix was proposed in 1980 by Fuhrmann and Hautus [7], who used a construction analogous to that of Wyman and Sain [1] in their study of the transmission zeros of a transfer function.

We consider, here, systems of the form

$$\sum = (A,B,C,D(s)) ,$$

with k-linear maps $A : X \to X$, $B : U \to X$, $C : X \to Y$, and $D(s) \ : \Omega U \to \Omega Y$. The system $\sum$ has transfer function

$$G(s) = C(sI - A)^{-1} B + D(s) .$$

The system matrix of $\sum$ is understood to be

$$S = \begin{bmatrix} sI-A & B \\ -C & D(s) \end{bmatrix} ,$$

which as a polynomial matrix admits a variety of possible module-theoretic interpretations. We examine some of these in the sequel.

FIVE ZERO MODULES

The following discussion may be regarded as a continuation of [1]; and familiarity with the first three sections of that work would be helpful in providing motivation for some of the subsequent steps. Because of space limitations, however, we cannot include all those ideas at this stage.

While the results of [1] considered strictly proper transfer functions and minimal realizations, the present discussion deals with a system which is not necessarily minimal, and allows the transfer function to have an improper part. The resulting generality suggests the possibility of considering several different, but closely related, zero modules for a system.

The transmission-zero module of the transfer function $G(s)$, defined in [1], is given by

$$Z(G) = \frac{G^{-1}(\Omega Y) + \Omega U}{\ker G(s) + \Omega U} .$$

Now denote by X the pole module of the system, analogous to the discussion, for example, in [1] and [8]; and let $\tilde{B} : \Omega U \to X$ and $\tilde{C} : X \to \Gamma Y$ be the usual factors of the Kalman input/output map. Then the (Rosenbrock) input-decoupling and output- decoupling zero modules are given by

$$Z_{i.d.} = X/\operatorname{im} \tilde{B} ,$$

$$Z_{o.d.} = \ker \tilde{C} .$$

It follows that $Z_{i.d.}$ is the module of equivalence classes of uncontrollable states and that $Z_{o.d.}$ is the module of unobservable states. These modules can be shown to have elementary divisors which coincide with the polynomials in [4, p. 64]. So the terminology is reasonable.

The system matrix S defines three quite different but closely related mappings, namely

(1) The $k(s)$-linear map $S : X(s) \oplus U(s) \to X(s) \oplus Y(s)$,

(2) the $k[s]$-linear map $S_\Omega : \Omega X \oplus \Omega U \to \Omega X \oplus \Omega Y$,

(3) the $k[s]$-linear map $S_\Gamma : \Gamma X \oplus \Gamma U \to \Gamma X \oplus \Gamma Y$.

We define two different "internal" zero modules by

$$Z_\Omega = \operatorname{coker} S_\Omega \; ; \; Z_\Gamma = \ker S_\Gamma .$$

The goal of this paper is to discuss the structure of these modules in detail and to relate them to the $Z_{i.d.}$, $Z_{o.d.}$, and $Z(G)$.

DISCUSSION OF RESULTS

An important technical result involves the study of S on the $k(s)$-vector space level: in fact there are $k(s)$-vector space isomorphisms

$$\ker G(s) \cong \ker S \; ; \; \operatorname{coker} G(s) \cong \operatorname{coker} S .$$

In particular, S is monic (epic) if, and only if, $G(s)$ is monic (epic).

These isomorphisms give two useful corollaries:

1. The Ω-zero module Z_Ω is finite-dimensional over k if, and only if, $G(s)$ is epic.

2. The Γ-zero module Z_Γ is finite-dimensional over k if, and only if, G(s) is monic.

Because of the simplifications afforded by these corollaries, we begin by considering right-invertible (epic) or left-invertible (monic) transfer functions.

Suppose that $\sum = (A,B,C,D(s))$ has a right-invertible transfer function. Then Z_Ω is finite-dimensional (i.e., a finitely generated torsion module). It turns out that Z_Ω contains all information about input-decoupling and transmission zeros, but that in general some information about output-decoupling zeros may be lost. More precisely, there is a submodule Z_1 of Z_Ω with corresponding factor module $Z_\Omega/Z_1 \cong Z_{i.d.}$. In fancier language, we have an exact sequence

$$0 \to Z_1 \to Z_\Omega \to Z_{i.d.} \to 0 .$$

The module Z_1 can be analyzed further; and there is another exact sequence

$$0 \to Z_2 \to Z_1 \to Z(G) \to 0 .$$

By now, both $Z_{i.d.}$ and $Z(G)$ have been "found" in Z_Ω. If G(s) is also left-invertible, then Z_2 is the set of controllable but unobservable states and is a well understood submodule of $Z_{o.d.}$. However, if G(s) is not monic, then Z_2 is more difficult to compute, and examples show that $Z_{o.d.}$ may make no contribution to Z_Ω at all!

If G(s) is left-invertible, then the Γ-zero module Z_Γ is a finitely-generated torsion module; and it contains a submodule isomorphic to $Z_{o.d.}$. The factor module $Z_{signal} \cong Z_\Gamma/Z_{o.d.}$ is closely related to the 1974 work of Desoer and Schulman [9]. See also Wyman and Sain [10]. Z_{signal} contains the transmission zero module $Z(G)$; and its factor module can be related to $Z_{i.d.}$; the situation is summarized by two exact sequences

$$0 \to Z_{o.d.} \to Z_\Gamma \to Z_{signal} \to 0$$

$$0 \to Z(G) \to Z_{signal} \xrightarrow{\alpha} Z_2' ,$$

where Z_2' is, intuitively, the module of observable but uncontrollable states. The map α is epic if G(s) is invertible, but not in general. Therefore, in some cases Z_Γ loses information about the i.d. zeros.

The situation is more complicated in the general case in which G(s) is neither monic nor epic. In general, both Z_Ω and Z_Γ are infinite dimensional. The Ω-zero module may have a non-zero free part, and the Γ-module may have a non-zero divisible part. However, there is a common finite-dimensional module called the "lumped zeros", and we can write

$$Z_\Omega \cong \text{(free module)} \oplus Z_{lumped} ,$$

$$Z_\Gamma \cong \text{(divisible module)} \oplus Z_{lumped} .$$

The module Z_{lumped} contains precisely the information about zeros which can be studied by numerical methods. Unfortunately this module is not as "well behaved" theoretically as Z_Ω and Z_Γ.

For example, although $Z_{i.d.}$ is a factor module of Z_Ω, it is not in general a factor of Z_{lumped}. There is an exact sequence

$$0 \longrightarrow (Z_1)_{\text{tors}} \longrightarrow Z_{\text{lumped}} \xrightarrow{j_1} Z_{\text{i.d.}} ;$$

but j_1 is not necessarily epic. To compute the precise image of j_1, we can use $C(sI-A)^{-1}$ to define a $k[s]$-module map

$$\tilde{C}_1 : Z_{\text{i.d.}} \longrightarrow Y(s)/(G(s)U(s) + \Omega Y)$$

such that $\operatorname{im} j_1 = \ker \tilde{C}_1$.

On the Γ-zero module side, we can no longer conclude that $Z_{\text{o.d.}}$ is a submodule of Z_Γ. In the general case we have a mapping $i_1 : Z_{\text{o.d.}} \longrightarrow Z_{\text{lumped}}$ such that

$$\ker i_1 = \tilde{B}(\ker G(s) \cap \Omega U) \cap Z_{\text{o.d.}} .$$

Thus, i_1 maps a given output decoupling zero to zero in Z_{lumped} exactly when it can be reached with an input from $\ker G(s) \cap \Omega U$. (Recall that output decoupling zeros are identified with states in the kernel of the observability map.)

Finally, we can report that Z_{lumped} is well-behaved with respect to the transmission zeros. The transmission zero module is a factor module of a submodule of Z_{lumped}. Formally, there are two exact sequences

$$0 \to (Z_1)_{\text{tors}} \to Z_{\text{lumped}} \to Z_{\text{i.d.}}$$

$$0 \to Z_2 \to (Z_1)_{\text{tors}} \to Z(G) \to 0 .$$

CONCLUSIONS

Despite an extraordinarily successful role in system theory, especially in relation to the meaning of various types of zeros, the meaning of the Rosenbrock system matrix in a module-theoretic sense remains a subject of research. In this paper, we propose three possible meanings for the system matrix, and two natural candidates for zero modules based upon these meanings. The discussion brings out numerous issues related to the definitions and suggests additional ideas for further study.

REFERENCES

1. Bostwick F. Wyman and Michael K. Sain, "The Zero Module and Essential Inverse Systems", IEEE Transactions on Circuits and Systems, Volume 28, Number 2, February 1981, pp. 112-126.

2. M.K. Sain, "The Growing Algebraic Presence in Systems Engineering: An Introduction", IEEE Proceedings, Volume 64, Number 1, January 1976, pp. 96-111.

3. Michael K. Sain, Introduction to Algebraic System Theory. New York: Academic Press, 1981.

4. H.H. Rosenbrock, State-Space and Multivariable Theory. New York: Wiley, 1970.

5. E.J. Davison and S.H. Wang, "Properties and Calculation of Transmission Zeros of Linear Multivariable Systems", Automatica, Volume 10,

Number 6, December 1974, pp. 643-658.

6. A.J. Laub and B.C. Moore, "Calculation of Transmission Zeros Using QZ Techniques", Automatica, Volume 14, Number 6, November 1978, pp. 557-566.

7. P.A. Fuhrmann and M.L.J. Hautus, "On the Zero Module of Rational Matrix Functions", Proceedings 19th IEEE Conference on Decision and Control, December 1980, pp. 256-257.

8. M.K. Sain, B.F. Wyman, R.R. Gejji, P.J. Antsaklis, and J.L. Peczkowski, "The Total Synthesis Problem of Linear Multivariable Control, Part I: Nominal Design", Proceedings 20th Joint Automatic Control Conference, Paper WP-4A, June 1981.

9. C.A. Desoer and J.D. Schulman, "Zeros and Poles of Matrix Transfer Functions and Their Dynamic Interpretation", IEEE Transactions on Circuits and Systems, Volume 21, Number 1, January 1974, pp. 3-8.

10. B.F. Wyman and M.K. Sain, "The Zero Module and Invariant Subspaces", Proceedings 19th IEEE Conference on Decision and Control, December 1980, pp. 254-255.

AN EFFICIENT POLE PLACEMENT BY OUTPUT FEEDBACK

S. Bingulac,
Dept. of Organizational Sciences,
University of Belgrade
Belgrade, Yugoslavia

J. Medanić,
Mihailo Pupin Institute,
Belgrade, Yugoslavia

ABSTRACT

A simple and computationally efficient algorithm for pole placement by output feedback is described. The approach consists of two stages. In the first stage m-1 poles are placed by a full rank regulator $u_1 = F_1 y$ while in the second stage these poles are kept invariant and additional k poles are placed at desired locations by a unity rank regulator $u_2 = F_2 y$, m and k being dimensions of input and output vectors respectively. An illustrative example is included.

1. PROBLEM FORMULATION

Consider a realization {A,B,C,} defining an open-loop multivariable dynamic system

$$\dot{x} = Ax + Bu \qquad (1)$$
$$y = Cx$$

with $x \varepsilon R^n$, $u \varepsilon R^m$, $y \varepsilon R^k$, rank $|B| = m$, rank $|C| = k$. Assume for the moment that $n = m+k-1$, and without loss of generality let $C = |I_k \vdots 0|$ and $k > m$.

The pole-placement problem may be formulated as follows: Find an output feedback regulator matrix $F \varepsilon R^{mxk}$, defining the control

$$u = Fy + v \quad (v \varepsilon R^m \text{ being an additional input}) \qquad (2)$$

such that all n poles of the closed-loop system matrix

$$H = A + BFC \qquad (3)$$

are equal to a given set of n real, or complex conjugate numbers,

$$\lambda_1^f, \ldots, \lambda_n^f, \qquad (4)$$

Many references have treated the output feedback pole placement problem (see Topaloglu and Seborg (1975), Paraskevapoulos (1976), Monro (1978) and Hopkins et al. (1981) and references therein), and a number of algorithms have been proposed for the numerical solution of the problem. The original proposals were based on unity rank feedback regulators (Jameson 1970, Davison 1970), and then on combinations of unity rank feedback regulators (Topaloglu and Seborg, 1975, Davison 1975), in order to preserve the linearity property in the conditions defining the solution of the problem. Unity rank solutions, however, have been shown to have poor robustness properties (Monro 1978, Beletruti 197), and therefore a number of other procedures have been proposed. In this paper a combination of methods is used to design a full rank output feedback regulator.

2. ALGORITHM DESCRIPTION

Assume for the moment that no eigenvalue λ_i^o of the open-loop system matrix A is equal to a given λ_j^f, i.e.

$$\lambda_i^o \neq \lambda_j^f \ ; \ i,j \varepsilon \{1,n\} \tag{5}$$

(The cases when the conditions $n = m+k-1$ and (5) are not satisfied are considered by Bingula and Medanic (1982). The pole placement procedure consists of the two stages. In the first stage m-1 poles are placed by a full rank feedback regulator F_1. In the second step the already placed m-1 poles are kept invariant and additional k poles are placed at desired locations (4) by a unity rank regulator F_2. The combined regulator F, of eq. (2), solving the formulated pole placement problem is given by $F=F_1+F_2$.

2.1 Placement of the m-1 poles

In accordance with the adopted pole placement procedure consider, instead of (2) and (3), the control

$$u = F_1 y + z, \quad z \varepsilon R^m \tag{6.a}$$

and the closed-loop system matrix

$$A_1 = A + B \ F_1 \ C. \tag{6.b}$$

Due to the structure of C, the product F_1C in (6.b) may be written as

$$F_1 C = F \cdot |I_k \vdots 0| = |F_1 \vdots 0| \triangleq F_c \tag{7}$$

Suppose T is an (n x n) matrix transforming A_1 into corresponding Jordan form H_j. Thus, using $H_j = T^{-1} A_1 T$ and defining

$$\tilde{F} \triangleq \tilde{F}_c \cdot T \triangleq |\underbrace{\tilde{F}_1}_{k} \vdots F_2| \ (m \times n), \tag{8}$$

eq. (6.b) becomes $T H_j - A T = B \tilde{F}$, $$\tag{9}$$

or $|t^1 \vdots \ldots \vdots t^n| \cdot H_j - A \cdot |t^1 \vdots \ldots \vdots t^n| = |\tilde{b}_f^1 \vdots \ldots \vdots \tilde{b}_f^n|; \ \tilde{b}_f^i \triangleq B \ \tilde{f}^i,$ (10)

with $\tilde{f}^i$ being the i-th column of the matrix $\tilde{F}$, i.e. $\tilde{F} \triangleq |\tilde{f}^1 \ldots \tilde{f}^n|$.

Assume that among the given closed-loop eigenvalues (4) there are:

$$\ell \text{ complex-conjugate pairs } \lambda_i^f \triangleq \delta_i \pm j \ w_i, \ i \varepsilon |1,\ell|, \text{ and} \tag{11}$$

$$n-2\ell \text{ real values } \qquad \lambda_i^f = \delta_i \qquad , \ i \varepsilon |2\ell+1,n|. \tag{12}$$

Using (10), the columns t^i of T may be calculated from

$$\begin{aligned} t^i &= Q_i^{-1} \ \tilde{b}_f^f \qquad\qquad \text{for } i \varepsilon |21+\ell, \ n|, \\ t^i &= |Q_j^2 + I_n w_j^2|^{-1} \ |Q_j \ \tilde{b}_f^i - w_j \ \tilde{b}_f^{i+1}| \\ & \qquad\qquad ; \ i \varepsilon |1,\ell| \ ; \ j \triangleq \left(\frac{i+1}{2}\right) \\ t^{i+1} &= |Q_j^2 + I_n w_j^2| \ |Q_j \ \tilde{b}_f^{i+1} + w_j \ b_f^i|; \text{ where } Q_j \triangleq (I_n \delta_j - A) \end{aligned} \tag{13}$$

Let us now restrict $\tilde{F}$ to have the structure

$$\tilde{F} = |\underbrace{F_1}_{k} \vdots \ 0| \ ; \quad 1 < \rho|\tilde{F}_1| < \min(m,k) = m \tag{14}$$

with $\tilde{F}_1$ arbitrary. Then, given the open loop realization {A,B,C} and the desired closed-loop poles λ_i^f, eq. (13) may be used to calculate k columns t^i for "almost" all F_1.

Although it is possible from (13) to calculate k columns t^i, the first stage of the described pole placement procedure requires that only m-1 columns be determined. (Note that $m-1 < k$).

Defining now the transformation matrix T^* as:

$$T^* \triangleq \left| t^1 \,\vdots\, \cdots \,\vdots\, t^{m-1} \,\vdots\, \begin{matrix} 0 \\ \hline I_k \end{matrix} \right| \quad (n \times n), \tag{15}$$

the matrix F_c, eq. (7), and the (m x k) feedback regulator matrix F_1 become

$$F_c = \tilde{F}(T^*)^{-1} = |\underbrace{F_1^*}_{m-1} \vdots\, 0| \quad (n \times m) \; ; \; F_1 = |\underbrace{F_1^*}_{m-1} \vdots\, 0| \quad (m \times k) \tag{16}$$

The detailed algorithm, referred to as Partial pole placement (ppp-Algorithm), that performs the calculations necessary to determine t_i and the matrix F_1 is presented in Bingulac and Medanić (1982).

From the definition of F_1 it may be concluded that the closed-loop system matrix

$$A_1 \triangleq A + B\, F_1 C \quad , \quad F_1 \triangleq |F_1^* \vdots\, 0| \quad (m \times k) \tag{17}$$

has m-1 poles equal to the first m-1 values from the set (5), while the remaining k poles of A_1 are arbitrary real or complex-conjugate numbers.

It should be noted that, in spite of the controllability of $\{A,B\}$, it may happen that, due to a particular choice of the matrix $\tilde{F}$, the columns t^i in (15) become linearly dependent. This problem may be eliminated by a slight perturbation in the elements $\tilde{f}_{ij}$ of $\tilde{F}$.

2.2 Placement of the remaining k desired poles

According to the development, presented in 2.1, it may be concluded that a similar procedure may be used for placing the remaining k desired poles. To this end a (m x k) feedback regulator matrix F_2, analogous to the matrix F_1, eq. (16), could be calculated. However, the calculated F_2 should not affect the already placed m-1 poles. To satisfy this requirement, in the closed-loop system,

$$\begin{aligned} \dot{x} &= A_1\, x + B\, z \quad ; \quad A_1 = A + B\, F_1 C \\ y &= C\, x \end{aligned} \tag{19}$$

obtained in the first stage (sec. 2.1), the poles already placed should be made uncontrollable. This is done using in principle the method suggested by Topaloglue and Seborg (1975). This may be achieved by considering the following single input system

$$\begin{aligned} \dot{x} &= A_1 + b\, z \\ y &= C\, x \end{aligned} \tag{20}$$

where the input vector b is given by

$$b = B\, g, \tag{21}$$

while the m-dimensional vector g should satisfy the following condition

$$T_j^{-1}\, B\, g = h \triangleq \left| \begin{array}{c} o \\ \vdots \\ o \\ \hline x \\ \vdots \\ x \end{array} \right| \begin{array}{l} \left.\vphantom{\begin{array}{c} o \\ \vdots \\ o \end{array}}\right\} m-1 \\ \left.\vphantom{\begin{array}{c} x \\ \vdots \\ x \end{array}}\right\} r \end{array} \quad (n \times 1) \tag{22}$$

In (22), T_J is the modal matrix transforming the matrix A_1 into the corresponding Jordan form H_{1J}.

It is assumed that the first m-1 columns in T_J, i.e. the first m-1 rows in T_J^{-1}, correspond to the m-1 poles assigned in the first stage of the pole placement procedure, (sec. 2.1).

It may be verified that the vector g appearing in (21) should belong to the null space of $Q = \hat{T}_J^{1}B$, i.e. should satisfy the condition

$$Q\ g = 0 \tag{23}$$

$\hat{T}_J^{1}$ being a (m-1) x n matrix containing the first m-1 rows from the T_J^{-1}.

A computational algorithm calculating the vector b without the necessity of calculating the Jordan form is given in Bingulac, Medanic (1982).

Thus, in the second stage of the proposed pole placement procedure the following single-input system is considered:

$$\begin{aligned} \dot{x} &= A_1\ x + b\ z \\ y &= C\ x \end{aligned} \tag{24}$$

In the output feedback control

$$z = f^2\ y + v \tag{25}$$

the output feedback regulator f^2

$$f^2 = \left| f_1^2 \vdots \dots \vdots f_k^2 \right| \ , \ (1 \times k) \tag{26}$$

should be selected in order to place the remaining k poles from (5). Since the matrix A_1 in (24), corresponds to the closed-loop matrix A_1, obtained in the first stage, eq. (17), from eqs. (21) - (23) it may be concluded that in the pair $\{A_1, b\}$ the already placed poles are uncoltrollable, i.e. cannot be affected by the control (25).

According to the development given in the sec. 2.1, it may be concluded that the n-dimensional columns t^i, determined from

$$t^i = (I_n \lambda_{m-1+i} - A_1)^{-1}\ b\ \tilde{f}_i^2 \ \ ; \ \ i\varepsilon|1,k|, \tag{27}$$

λ_{m-1+i} being desired poles from (5), may be used in obtaining the regulator f^2. In other words, the regulator f^2 is given by

$$f^2 = \tilde{f}^2\ \hat{T}^* \tag{28}$$

where $\hat{f}^2 \triangleq \left| \tilde{f}_1^2 \vdots \dots \vdots \tilde{f}^2 \right|$ is k-dimensional row vector with "almost" arbitrary elements $\tilde{f}_i^2$, while the (k x k) matrix $\hat{T}^*$ is determined from the columns t^i, i.e.

$$\hat{T}^* = T_1^{-1} \ ; \left| \begin{array}{c} T_1 \\ \hline T_2 \end{array} \right| \}k = \left| t^1 \vdots \dots \vdots t^k \right|, \tag{29}$$

t^i, $i\varepsilon|1,k|$, being defined by (27).

Finally, returning back to (24), (25) and using (21), the (m x k) output feedback regulator matrix F_2, placing the remaining k poles to the desired locations (5) may be expressed by

$$F_2 = g\ f^2 \tag{30}$$

2.3 Regulator Structure

From the development explained in 2.1 and 2.2, it may be concluded that the feedback regulator matrix F satisfying (3) and (4), i.e. solving the formulated pole placement problem, Fig. 1, is given by: $F = F_1 + F_2 = F_1 + g\, f^2$. Note that $\rho|F_1| = m-1$ and $\rho|F_2| = 1$, while, according to (2) it may be expected that in almost all cases the regulator F would be of the full rank, i.e. $F = \min(m,k) = m$.

The complete pole placement algorithm described in the paper is presented in Bingulac, Medanic (1982), while the block diagram of the pole placement procedure is given in Fig. 1.

3. COMPUTATIONAL EXAMPLE

As an illustrative example, consider the following open-loop realization $\{\tilde{A},\tilde{B},\tilde{C}\}$ with n=5, m=3 and k=3:

$$\tilde{A} = \begin{vmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ -5 & -10 & -7 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & -13 & -6 \end{vmatrix} ; \quad \tilde{B} = \begin{vmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{vmatrix} ; \quad \tilde{C} = \begin{vmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{vmatrix} \qquad (31)$$

Transforming the realization $\{\tilde{A},\tilde{B},\tilde{C}\}$ into the required form $\{A,B,C\}$ with $C = |I_k \vdots 0|$, one obtains for A and B:

$$A = \begin{vmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ -5 & -10 & 1 & -7 & 0 \\ 0 & 0 & -13 & 0 & -6 \end{vmatrix} ; \qquad B = \begin{vmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{vmatrix} . \qquad (32)$$

Let the desired poles of the closed-loop system to be placed by the feedback regulator be:

$$-1 \pm j1, \; -2, \; -3 \pm j2 \, . \qquad (33)$$

Since, m=3, in the first stage of the pole placement procedure only two poles will be placed. Let these poles be $-1 \pm j1$.

The first stage of the algoritm, eq. (13) requires an auxiliary and arbitrary $|m \times (m-1)|$ matrix $\tilde{F}_1$. Let

$$\tilde{F}_1 = \begin{vmatrix} 1 & 1 \\ 1 & 2 \\ 1 & 3 \end{vmatrix} \qquad (34)$$

Using $A, B, \tilde{F}_1$ and the poles (33), the PPP-algorithm renders for the output feedback regulator, eq. (16)

$$F_1^* = \begin{vmatrix} -0.647 & -0.247 \\ -0.895 & -0.170 \\ -1.142 & -0.093 \end{vmatrix} \qquad (35)$$

The closed loop system matrix A_1, eq. (17), obtained after the first stage, see Fig. 1, becomes: $A_1 = A + B F_1 C = A + B\, |F_1^* \vdots 0| =$

$$A_1 = \begin{vmatrix} 0.0 & 1.0 & 0.0 & 0.0 & 0.0 \\ -0.647 & -0.247 & 0.0 & 1.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 1.0 \\ -5.895 & -10.170 & 1.0 & -7.0 & 0.0 \\ -1.142 & -0.093 & -13.00 & 0.0 & -6.0 \end{vmatrix} \qquad (36)$$

The poles of A_1 are

$$\begin{array}{lrrrrr} \text{real parts} & -1.0 & -1.0 & -5.23 & -3.107 & -3.107 \\ \text{imag. parts} & 1.0 & -1.0 & 0.0 & 2.100 & -2.100 \end{array} \qquad (37)$$

The second stage of the pole placement procedure starts with the calculation of the m-dimensional vector g, e.q. (20) - (24), which will guarantee that in the single-input system $\{A_1, b^1\}$; $b^1 \triangleq B\, g$, the alreadly placed poles $-1 \pm j1$ are uncontrollable. The vector g, becomes

$$g = \begin{vmatrix} 6.153 \\ -0.477 \\ 1.000 \end{vmatrix} \quad , \text{ leading to } \quad b = \begin{vmatrix} 0.000 \\ 6.153 \\ 0.000 \\ -0.477 \\ 1.000 \end{vmatrix} \qquad (38)$$

The second stage of the algorithm, eq. (27), requires an auxiliary and arbitrary k-dimensional row $\tilde{f}^2$. Let:

$$\tilde{f}^2 = |1 \quad 2 \quad 3| . \qquad (39)$$

Using A_1, b^1, f^2 and the remaining poles to be placed, i.e. -2, $-3 \pm j2$, the algorithm yields the feedback regulator f^2 as:

$$f^2 = |38.53 \quad 52.75 \quad 6.614| \qquad (40)$$

Thus, the complete output-feedback regulator F, becomes

$$F = |F_1^* \quad 0| + g\, f^2 = \begin{vmatrix} 1.723 & 2.998 & 0.406 \\ -19.271 & -25.329 & -3.152 \\ 37.387 & 52.657 & 6.610 \end{vmatrix} \qquad (41)$$

which guarantees that the poles of the closed-loop system

$$A + B\, F\, C = A + B \quad |F \quad 0| =$$

$$\begin{vmatrix} 0.000 & 1.000 & 0.000 & 0.000 & 0.000 \\ 1.723 & 2.998 & 0.406 & 1.000 & 0.000 \\ 0.000 & 0.000 & 0.000 & 0.000 & 1.000 \\ -24.271 & -35.329 & -2.152 & -7.000 & 0.000 \\ 37.387 & 52.657 & -6.389 & 0.000 & -6.000 \end{vmatrix}$$

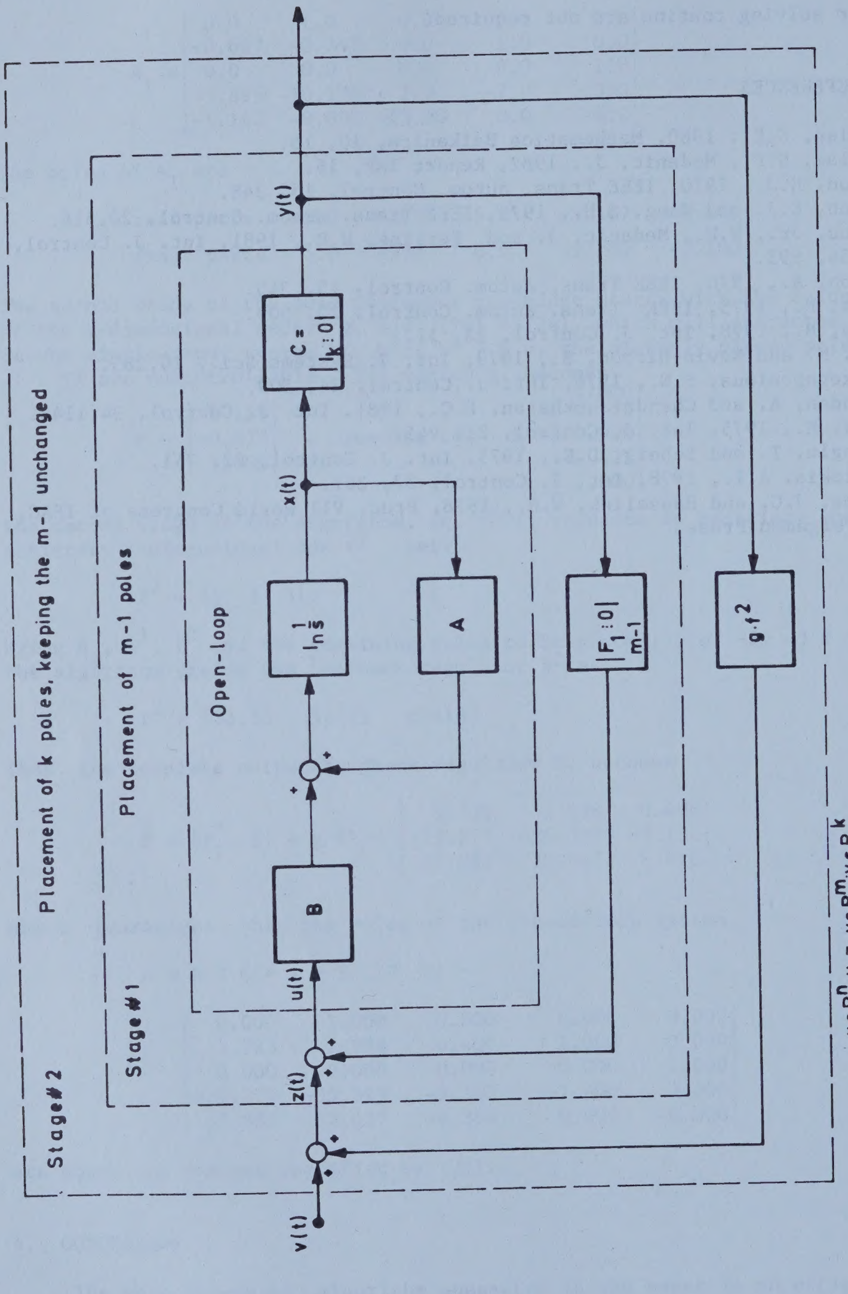
are equal to the set specified by (33).

4. CONCLUSION

The pole placement algorithm suggested in the paper is an efficient method for synthesis of full rank output feedback regulator to place all poles of the given system at desired locations. The algorthm is applicable to the design of both static and dynamic regulators depending on the number of system inputs and outputs with respect to the order of the state vector. The numerical steps are straightforward and rely basically on the use of reliable matrix inversion procedures, while eigenvalue and eigenvector solving routine are not required.

5. REFERENCES

Bingulac, S.P., 1980, Mathematica Balkanica, 10, 15.
Bingulac, S.P., Medanic, J., 1982, Report IMP, 16.
Davison, E.J., 1970, IEEE Trans. autom. Control, 15, 348.
Davison, E.J. and Wang, S.H., 1975, IEEE Trans. autom. Control, 20,516.
Hopkins, Jr., W.H., Medanic, J. and Perkins, W.R., 1981, Int. J. Control, 34, 593.
Jameson, A., 1970, IEEE Trans. autom. Control, 15, 345.
Kimura, H., 1975, IEEE Trans. autom. Control, 20, 509.
Kimura, H., 1978, Int. J. Control, 28, 11.
Munro, N. and Novin-Hirbod, S., 1979, Int. J. Systems Sci., 10,285.
Paraskevopoulous, P.N., 1976, Int. J. Control, 24, 509.
Sambandan, A. and Chandarasekharan, P.C., 1981, Int. J. Control, 34,1143.
Seraji, H., 1975, Int. J. Control, 21, 945.
Topaloglu, T. and Seborg, D.E., 1975, Int. J. Control, 22, 741.
Vardulakis, A.I., 1978, Int. J. Control, 27, 261.
Willems, J.C. and Hasselink, W.H., 1978, Proc. VII World Congress of IFAC, Pergamon Press.


Stage#2
Placement of k poles, keeping the m-1 unchanged
Stage#1
Placement of m-1 poles
Open-loop
v(t)
z(t)
u(t)
B
$I_n \frac{1}{s}$
x(t)
A
$C = |I_k : 0|$
y(t)
$|F_1 : 0|$
m-1
$g \cdot f^2$
+
+
+
+
+
+

$x \in R^n; u, z, v \in R^m, y \in R^k$
$k > m$
$n = m + k - 1$

Fig. 1

TOWARDS A FUZZY GEOMETRIC THEORY OF LINEAR MULTIVARIABLE CONTROL

HEBERTT SIRA RAMIREZ
Control Systems Department, Universidad de Los Andes, Merida-Venezuela

ABSTRACT

This paper develops a generalization of Wonham's Geometric Theory of Linear Multivariable Control [1] by introducing a fuzzy subspace formulation into the conventional theory. This generalization allows for treatment, within a convenient mathematical framework, of inescapable issues in systems analysis and design such as : vagueness or imprecision in control restrictions , disturbance description and design objectives. We furnish an appropiate fuzzy geometric basis for the restatement and solution development of problems treated by the conventional Geometric Theory for the control of Linear Multivariable Systems.

I INTRODUCTION

The purpose of this paper is to introduce and develop the fuzzy set theoretic context within the Geometric Theory of Linear Multivariable Control introduced by Wonham [1]

At the very heart of this generalization is the concept of fuzzy subspace of a real vector space. Its introduction allows for the convenient and mathematically tractable modeling of :subjective or vague control restrictions, imprecise disturbance knowledge and fuzzy design objectives. Statements and requirements expressing a subjective knowledge of systems variables and design specifications can be adequately modeled by this technique in a way that enlarges the solution possibilities even from a practical viewpoint. This power is not sheard by the conventional "crisp" theory.

It is noteworthy that fuzzy set theory, invented by a control theorician [2], has rapidly evolved in directions out of the control systems discipline (Cognitive Processes, Grammars, Graph Theory, Risk Analysis etc [3]) while questions related to utilizing fuzzy sets or fuzzy subspaces as a modeling device for describing poorly defined variables, signals and objectives, have only been partially explored [4]-[7].

Geometric theory, on the other hand, has reached a state of maturity and rapid growth bridging, day after day, with old and recent areas within the control theory arsenal. Recent contributions, especially by Willems [8]-[13] , point to the necessity of relaxing some of the basic concepts in Geometric Theory so that some degree of freedom and approximative characteristics are introduced in the formulation of problems succesfully treated by the Geometric Theory of Wonham. Our viewpoint touches, tangentially, on this phylosophical necessity and conicides, although in a non-technical manner, with the theory developed by Willems of " Almost Controlled Invariant and Almost Conditionally Invariant Subspaces ".

This article presents in its section II all the basic definitions and results related to the proposed generalization. We introduce there the definitions of : fuzzification of a linear vector space (subspace), fuzzy subspace ,fuzzy images and fuzzy kernels, fuzzy invariant subspaces and fuzzy factor spaces. These definitions allow, in turn, for the corresponding generalization,to the fuzzy context, of concepts such as reachable subspaces, (A,B)-invariant (or controlled invariant, or A-mod $\mathcal{B}$ invariant) subspaces ,(A,C) -invariant (or conditionally invariant, or $A \mid \operatorname{Ker} C$ - invariant) subspaces, unobservable subspaces and controllability subspaces. All these definitions, though, are based in the elementary fuzzy-set operations of vector addition of fuzzy subspaces, intersection of fuzzy subspaces etc. We also present formulae needed in the algorithms that produce fuzzy maximal invariant subspaces, maximal reachability and maximal unobservable fuzzy subspaces within a given fuzzy subspace.

In essence, all the variety of problems treated by the Geometric Theory of Linear Multivariable Control could be restated in terms of the fuzzy-geometric generalization. Nevertheless, we shall only formulate one such problem; the Fuzzy Disturbance Rejection Problem (FDRP), in section III of this work, as an example of the possibilities of the proposed generalization.

All the background material about fuzzy sets, and some basic operations on these sets, as presented in [6], are collected at the end of the article in an appendix. The work also presents some conclusions and suggestions for further research in this area.

II NOTATION AND BASIC DEFINITIONS

In this section we introduce the basic definitions needed throughout the entire article. These definitions are centered around the idea of fuzzy subspaces. We introduce this concept by simply adscribing a membership function (m.f) defined on the support subspace. This concept joined to all known elementary operations on fuzzy sets permits the "reconstruction" of the Geometric Theory with a fuzzy-set theoretic context.

Definition 1 A fuzzy vector space (subspace) is defined as a pair $(\mathcal{U},\mu)$, usually denoted by $\mathcal{U}_\mu$, where $\mathcal{U}$ is a vector space (subspace) and μ is a membership function (m.f) (i.e, a map:$\mathcal{U} \to [0,1]$) defined on $\mathcal{U}$. We shall refer to $\mathcal{U}$ as the support space (subspace) of $\mathcal{U}_\mu$. Simbolically, we express this fuzzy space (subspace) by means of Zadeh's notation as :

$$\mathcal{U}_\mu = \int_{\mathcal{U}} \mu(\underline{v}) / \underline{v} \qquad (2.1)$$

A ν-fuzzification of $\mathcal{X}$ is understood as the adscription of a m.f. ν to a subspace (or space) $\mathcal{X}$. When confussion could develop, we denote a m.f. ν adscribed to a subspace $\mathcal{X}$ as $\nu_{\mathcal{X}}$.

Definition 2 Let B be a linear map $B : \mathcal{U} \to \mathcal{X}$ and let $\mathcal{U}_\mu$ be a fuzzy subspace defined by a μ-fuzzification of $\mathcal{U}$. The fuzzy image of B or the μ-image of B in $\mathcal{X}$ is determined by :

$$\mathcal{B}_\mu = \{ \underline{x} \in \mathcal{X} : \ \underline{x} = B\,\underline{u} \ \text{ for some } \underline{u} \in_f \mathcal{U} \} \qquad (2.2)$$

and denoted by $\mathcal{B}_\mu$. $\mathcal{B}_\mu$ is then a fuzzy subspace whose support subspace is $\mathcal{B}$ and its m.f. is inherited from the fuzzification of $\mathcal{U}$. According to the extension principle (Zadeh [2]) we have :

$$\mathcal{B}_\mu = \int_{\mathcal{B}} \mu(B^{-1}\underline{x}) / \underline{x} \qquad (2.3)$$

We shall often use the composition symbol to express the m.f. inheritance through certain linear map of a m.f. adscribed to an original space. In the preceeding case we would write :

$$\mu_{\mathcal{B}} = \mu \circ B^{-1} \qquad (2.4)$$

Definition 3 Let $\mathcal{Y}_\delta$ denote a δ-fuzzification of $\mathcal{Y}$, by means of the m.f. δ , representing the qualitative statement : " $\mathcal{Y}_\delta$ is the set of vectors very close to zero in $\mathcal{Y}$ " i.e $\delta(0) = 1$ while $\delta(\underline{y}) \simeq 0 \ \forall \ \underline{y} \neq \underline{0}$. We call $\mathcal{Y}_\delta$ a "fuzzy zero" and it is usually denoted, in this work, by $\underline{0}_\delta$.

Definition 4 Let C be a linear map $C : \mathcal{X} \to \mathcal{Y}$, then the fuzzy kernel of C , denoted by $\text{Ker}_\delta\, C$ or δ-ker C, is the fuzzy subspace :

$$\text{Ker}_\delta\, C = \{ \underline{x} : \ C\,\underline{x} \in_f \underline{0}_\delta \} \qquad (2.5)$$

The m.f. of this fuzzy subspace is inherited from that of the fuzzy zero. By virtue, again, of the extension principle we have :

$$\delta_{\text{Ker } C} = \delta \circ C \qquad (2.6)$$

Definition 5 Let $\mathcal{V}_\mu$ be a fuzzy subspace with m.f. μ , we say $\mathcal{V}_\mu$ is an A-invariant fuzzy subspace if :

$$A\,\mathcal{V}_\mu \subset_f \mathcal{V}_\mu \qquad (2.7)$$

This condition implies two relations, one involving the support subspace and the other involving the m.f. :

$$AV \subset V \quad \text{and} \quad \mu_{AV} \leqq \mu \quad \text{or} \quad \mu_V \circ A^{-1} \leqq \mu_V \tag{2.8}$$

<u>Definition 6</u> Let R_μ and S_ν be fuzzy subspaces, we denote the fuzzy addition of these subspaces by $R_\mu + S_\nu$ and define it as :

$$R_\mu + S_\nu = \{ \underline{x} + \underline{y} : \underline{x} \, \varepsilon_f \, R_\mu \text{ and } \underline{y} \, \varepsilon_f \, S_\nu \} \tag{2.9}$$

where the sum is understood in a fuzzy set-theoretic way. Adopting the definition of fuzzy addition given in [6] (See also the Appendix) we have :

$$R_\mu + S_\nu = \int_{R+S} \sup_{\underline{v}} [\mu(\underline{u}-\underline{v}) \wedge \nu(\underline{v})] / \underline{u} \tag{2.10}$$

We shall express shortly the m.f. of this sum as :

$$\zeta_{R+S} = \mu_R * \nu_S \tag{2.11}$$

or :

$$R_\mu + S_\nu = (R + S)_{\mu * \nu} \tag{2.12}$$

where the symbol * stands for <u>convolution</u> of the involved m.f.'s according to the formula (2.10) above.

<u>Definition 7</u> Let V_μ and R_ν be two fuzzy subspaces, then the intersection of thses subspaces is a fuzzy subspace defined as :

$$V_\mu \cap R_\nu = (V \cap R)_{\mu \wedge \nu} \tag{2.13}$$

<u>Definition 8</u> Let $A : X \to X$ be a linear map, not necessarily invertible, and let S_μ be a fuzzy subspace of X . <u>The μ -inverse image of S under A</u> is defined as the fuzzy subspace given by :

$$A^{-1} S_\mu = \{ \underline{x} : A \underline{x} \, \varepsilon_f \, S_\mu \} \tag{2.14}$$

According to the extension principle we have :

$$A^{-1} S_\mu = (A^{-1} S)_{\mu \circ A} \tag{2.15}$$

<u>Definition 9</u> Let X_μ be a fuzzy subspace and $S_\mu \subset_f X_\mu$ a fuzzy subspace of X_μ. Then, we denote by X_μ / S_μ the factor space (or subspace) of X_μ modulo S_μ. (also denoted by X_μ (mod S_μ). The support subspace is X/S and it constitutes a vector subspace with elements denoted by either $[\underline{x}]$ or $\underline{x}(\text{mod}\, S)$.

Using the definition of equivalence classes as the sum of a singleton and a subspace, we can, after using the formula for <u>fuzzy translation of a fuzzy set</u> (in this case of a fuzzy subspace), determine the fuzzy factor space as the fuzzy subspace : (See the Appendix)

$$X_\mu / S_\mu = \int_{X/S} \sup_{\underline{u}} \{ \bigvee_{\underline{y} \varepsilon \underline{x} + S} [(\mu_X | S)(\underline{u}-\underline{y}) \wedge \mu_X(\underline{y})] \} / [\underline{x}] \tag{2.16}$$

In conventional Geometric Control Theory, when one defines the reachable or controllable subspace $< A|B >$, use is made of the fact that $\underline{u}(t)$, the control vector, can take <u>any</u> value in the set of allowable control functions over the infinite time interval. It is not difficult to see that a μ-fuzzification of the control space U yields a fuzzification of the reachable subspace through the μ-image of the control channel map B and the systems dynamics. It is left to the reader show that actually the subspace of fuzzy reachable states in this case is given by :

$$R_\mu \triangleq <A| B_\mu> = B_\mu + AB_\mu + A^2 B_\mu + \ldots + A^{n-1} B_\mu \tag{2.17}$$

and then :

$$\mu_{<A|B_\mu>} = \mu_B * \mu_{AB} * \ldots * \mu_{A^{n-1}B} = \mathop{*}_{i=0}^{n-1} \mu_B \circ A^{-i} \tag{2.18}$$

<u>Definition 10</u> A fuzzy subspace X_ν in R^n is said to be μ-controllable or μ-reachable if and only if : $< A|B_\mu > \supset_f X_\nu$ i.e it is simmultaneously satisfied :

$$\langle A \mid \mathcal{B} \rangle \supset \mathcal{X} \quad \text{and} \quad \mu_{\langle A \mid \mathcal{B} \rangle} \geqq \nu_{\mathcal{X}} \tag{2.19}$$

Definition 11 $\mathcal{V}_\nu$ is $(A,B)_\mu$-invariant (also called $A(\mathrm{mod}\,\mathcal{B}_\mu)$-invariant or Controlled Invariant subspace) if there exists a map F_μ such that $\mathcal{V}_\nu$ is $(A+BF_\mu)$-invariant, i.e $(A + BF_\mu)\mathcal{V}_\nu \subset_f \mathcal{V}_\nu$.

As a counterpart to Wonham's theorem on (A,B)-invariant subspaces, we also have :

Theorem 1 $\mathcal{V}_\nu$ is $(A,B)_\mu$ -invariant subspace if and only if :

$$A\,\mathcal{V}_\nu \subset_f \mathcal{V}_\nu + \mathcal{B}_\mu \tag{2.20}$$

In terms of the support subspace and the m.f.'s this means :

$$A\,\mathcal{V} \subset \mathcal{V} + \mathcal{B} \tag{2.21}$$

and

$$\nu_{\mathcal{V}} \circ A^{-1} \leqq \nu_{\mathcal{V}} * \mu_{\mathcal{B}} \tag{2.22}$$

Definition 12 A fuzzy subspace S_ν is said to be $(C,A)_\delta$-invariant (also called $A \mid \mathrm{Ker}_\delta C$ -invariant or Conditionally Invariant subspace) if and only if

$$A(\, S_\nu \cap \mathrm{Ker}_\delta\, C \,) \subset_f S_\nu \tag{2.23}$$

this condition implies the simultaneous verification of :

$$A(\, S \cap \mathrm{Ker}\, C \,) \subset S \tag{2.24}$$

and

$$[\, \nu_S \wedge (\, \delta \circ C) \,] \circ A^{-1} \leqq \nu_S \tag{2.25}$$

We now make use of the fuzzy kernels, in order to study fuzzy unobservable subspaces.

Definition 13 A subspace Z_ζ is δ-unobservable if and only if :

$$Z_\zeta \subset_f \bigcap_{i=0}^{n-1} A^{-i}\, \mathrm{Ker}_\delta\, C \tag{2.25}$$

This condition implies :

$$Z \subset \bigcap_{i=0}^{n-1} A^{-i}\, \mathrm{Ker}\, C \tag{2.26}$$

and

$$\bigwedge_{i=0}^{n-1} [\, \delta \circ (\, C\, A^{i} \,)] \geqq \zeta \tag{2.27}$$

In conventional Geometric Theory of control systems, it is shown that the class of (A,B)-invariant subspaces of a vector space $\mathcal{N}$, denoted by $\mathcal{I}(A,B,\mathcal{N})$ or simply $\mathcal{I}(\mathcal{N})$ is an upper semilattice relative to subspace inclusion and addition. As such, if $\mathcal{I}(\mathcal{N})$ is non-empty, then it contains a unique supremal (A,B)-invariant subspace $\mathcal{V}^*$. This subspace can be computed in a finite number of steps according to the following recursive algorithm :

$$\mathcal{V}^* = \mathcal{V}^n ; \quad \mathcal{V}^i = \mathcal{N} \cap A^{-1}(\, \mathcal{B} + \mathcal{V}^{i-1} \,) \,,\ i \in \underline{n} \ ; \quad \mathcal{V}^0 = \mathcal{N} \tag{2.28}$$

Let $\mathcal{N}_\delta$ be a fuzzy subspace and R^m_μ a μ-fuzzification of R^m. The supremal (or largest) $(A,B)_\mu$-invariant subspace contained in $\mathcal{N}_\delta$ is obtained recursively by means of :

$$\mathcal{V}^*_{\psi^*} = \mathcal{V}^n_{\lim_{j\to\infty} \psi_j} \ ; \quad \mathcal{V}^i_{\psi_j} = \mathcal{N}_\delta \cap A^{-1}(\, \mathcal{B}_\mu + \mathcal{V}^{i-1}_{\psi_{j-1}} \,) \ ;\ i=j \text{ in } \underline{n},\ i=n,\ j \geqq n+1$$

with the "initial condtion" $\mathcal{V}^0_{\psi_0} = \mathcal{N}_\delta$ (2.29)

The above algorithm splits, as usual, in two algorithmic procedures ; one defined on the support subspaces and the other defined on the m.f's. The first one converges, necessarily, in n steps at the most. The second algorithm needs, in general, a larger number of iterations. For this reason , the sub-indices are equal in $\underline{n}$ but from the n-th step on, the sub-index i stops at n while the sub-index j continues growing until convergence of the m.f. The support subspace algorithm is the same given by (2.28) above,while the m.f. algorithm is represented by :

$$\psi^* = \lim_{j\to\infty} \psi_j \;;\; \psi_j = \delta \wedge [(\mu_{\mathcal{B}} * \psi_{j-1}) \circ A\,] \,,\; j=1,2,\ldots \;;\; \psi_0 = \delta \qquad (2.30)$$

The dual notion of $\mathcal{V}^*$, the smallest $A|\mathrm{Ker}\,C$ -invariant subspace (or (C,A)-invariant subspace) containing a given subspace $\mathcal{K}$, is denoted by $\mathcal{J}_*$ and the algorithm that produces this subspace is :

$$\mathcal{J}_* = \mathcal{J}^n \;;\; \mathcal{J}^i = \mathcal{K} + A\,[\,\mathcal{J}^{i-1} \cap \mathrm{Ker}\,C\,] \,,\; i \in \underline{n} \;;\; \mathcal{J}^0 = 0 \qquad (2.31)$$

The fuzzy version of this algorithmic procedure is stated as follows: Let $\mathcal{K}_\mu$ be a fuzzy subspace and 0_δ a fuzzy zero in R^n. Let us denote by $\partial(\underline{u})$ the Dirac unit impulse function centered at $\underline{u}$ i.e $\partial(\underline{u}) = 1$ and $\partial(\underline{v})=0$ $\forall\, \underline{v} \neq \underline{u}$. The infimal (smallest) $(C,A)_\delta$-invariant subspace containing $\mathcal{K}_\mu$ is obtained recursively by means of :

$$\mathcal{J}_{*\nu_*} = \mathcal{J}^n_{\lim_{j\to\infty}\nu_j} \;;\; \mathcal{J}^i_{\nu_j} = \mathcal{K}_\mu + A[\,\mathcal{J}^{i-1}_{\nu_{j-1}} \cap \mathrm{Ker}_\delta\, C\,] \,,\; i=j \text{ in } \underline{n};\; i=n,\, j \geq n+1$$

with the "initial condition" : $\mathcal{J}^0_{\nu_0} = \{0\}_{\partial(0)}$ (2.32)

i.e the m.f. algorithm would be given by :

$$\nu_* = \lim_{j\to\infty} \nu_j \;;\; \nu_j = \mu * \{[\nu_{j-1} \wedge (\delta \circ C\,)] \circ A^{-1}\},\, j=1,2\ldots \;;\; \nu_0 = \partial(0) \qquad (2.33)$$

One of the crucial concepts in Geometric Control Theory is that of a Controllability Subspace (C.S). The fuzzy version of these important subspaces is characterized by the statement given in the following definition.

Definition 14 A fuzzy subspace $\mathcal{R}_\nu$ is a μ-controllability subspace if there exists a matrix F_μ such that :

$$\mathcal{R}_\nu = \langle A + BF_\mu \,|\, \mathcal{B}_\mu \cap \mathcal{R}_\nu \rangle \qquad (2.34)$$

according to this the m.f. satisfies :

$$\nu = \mathop{*}_{i=0}^{n-1} [(\mu \wedge \nu) \circ (A+BF)^{-i}] \qquad (2.35)$$

The fundamental theorem which allows characterization of fuzzy C.S.'s is the following :

Theorem 2 Let A, B , $\mathcal{R}_\nu$ be fixed; $\mathcal{R} \subset \mathcal{X}$, $\mathcal{R}_\nu$ is a fuzzy C.S. of (A,B) if and only if $\mathcal{R}_\nu$ is $(A,B)_\mu$-invariant and $\mathcal{R}_\nu = \mathcal{S}_{*\nu_*}$ where $\mathcal{S}_{*\nu_*}$ is the infimal subspace such that:

$$\mathcal{S}_{*\nu_*} = \mathcal{R}_\nu \cap (\,A\mathcal{S}_{*\nu_*} + \mathcal{B}_\mu\,) \qquad (2.36)$$

furthermore:

$$\mathcal{S}_{*\nu_*} = \mathcal{S}^n_{\lim_{j\to\infty}\nu_j} \;;\; \mathcal{S}^i_{\nu_j} = \mathcal{R}_\nu \cap (A\,\mathcal{R}^{i-1}_{\nu_{j-1}} + \mathcal{B}_\mu) \;;\; i=j \text{ in } \underline{n} ,\, i=n,\, j \geq n+1$$

with the "initial condition" : $\mathcal{R}^0_{\nu_0} = \{\,0\}_{\partial(0)}$ (2.37)

i.e the support subspace satisfies the algorithm :

$$\mathcal{R} = \mathcal{S}_* \;;\; \mathcal{S}_* = \mathcal{R} \cap (\,A\mathcal{S}_* + \mathcal{B}\,) \;;\; \mathcal{S}^i = \mathcal{R} \cap (A\,\mathcal{S}^{i-1} + \mathcal{B}\,) \;;\; \mathcal{S}^0 = 0 \qquad (2.38)$$

while the m.f. satisfies :

$$\nu_* = \lim_{j\to\infty} \nu_j \;;\; \nu_j = \nu \wedge [(\nu_{j-1} \circ A^{-1}) * \mu_{\mathcal{B}}] \,,\; j=1,2,\ldots \;;\; \nu_0 = \partial(0) \qquad (2.38)$$

This procedure allows one to check whether a fuzzy subspace $\mathcal{R}_\nu$ is a fuzzy C.S.

Let R^m_μ be a fuzzification of R^m and let $\mathcal{V}^*_{\psi^*}$ be the largest $(A,B)_\mu$-invariant subspace contained in $\mathcal{K}_\delta$. Then, the supremal fuzzy controlability subspace algorithm (FCSA) which renders $\mathcal{V}^*_{\psi^*}$ is given by:

$$\mathcal{R}^*_{\xi^*} = \mathcal{R}^n_{\lim_{j\to\infty}\xi_j} \;;\; \mathcal{R}^i_{\xi_j} = \mathcal{V}^*_{\psi^*} \cap (\,\mathcal{B}_\mu + A\mathcal{R}^{i-1}_{\xi_{j-1}}\,) \,,\, j=i \text{ in } \underline{n} ,\, i=n,\, j \geq n+1 \qquad (2.39)$$

with the "initial condition" : $\mathcal{R}^0_{\xi_0} = \{0\}_{\partial(0)}$

The support subspace is generated recursively via:

$$\mathcal{R}^* = \mathcal{R}^n \; ; \quad \mathcal{R}^i = \mathcal{V}^* \cap (\mathcal{B} + A \, \mathcal{R}^{i-1}) , \; i \in \underline{n} \quad ; \quad \mathcal{R}^0 = 0 \tag{2.40}$$

while the m.f is computed by means of :

$$\xi_* = \lim_{j \to \infty} \xi_j \; ; \quad \xi_j = \psi_* \Lambda \, [\mu_{\mathcal{B}} * (\xi_{j-1} \circ A^{-1})] ; \; j=1,2,\ldots \; ; \; \xi_0 = \partial(0) \tag{2.41}$$

Another important concept in Geometric Control Theory of linear systems is that of detectability which turns out to be a property weaker than that of observability and closely related to it. (Wonham [1]). This property is established when it is found that, for a particular system, the subspace of unstable modes is observable.

Let the minimal polynomial (m.p) $\alpha(\lambda)$ of A be factored as $\alpha(\lambda) = \alpha^+(\lambda) \, \alpha^-(\lambda)$. where the zeros of $\alpha^+(\lambda)$ (respectively $\alpha^-(\lambda)$) in the complex plane lie in the closed right (respectively : open left) half plane. Let :

$$\mathcal{X}^+(A) = \operatorname{Ker} \alpha^+(A) \; ; \; \mathcal{X}^-(A) = \operatorname{Ker} \alpha^-(A) \tag{2.41}$$

a pair (C,A) is detectable if :

$$\bigcap_{i=0}^{n-1} A^{-i} \operatorname{Ker} C \subset \mathcal{X}^-(A) \tag{2.42}$$

i.e. A is stable on the unobservable subspace of (C,A).

Let $1_{\mathcal{X}^-}$ denote the membership function of the crisp set $\mathcal{X}^-(A)$ along the subspace indicated in the second formula of (2.41). This m.f. has the value zero elsewhere. Using this m.f. and adscribing it to $\mathcal{X}^-(A)$ we can propose the concept of a fuzzy detectable pair whenever, given a fuzzy zero characterized by the m.f. δ in the output space, it is verified that :

$$\bigwedge_{i=0}^{n-1} \delta \circ C A^i \leqq 1_{\mathcal{X}^-} \tag{2.43}$$

III FUZZY DISTURBANCE DECOUPLING PROBLEM

In this section we shall formulate the Fuzzy Disturbance Decoupling Problem (FDDP). This problem is concerned with decoupling in an approximate (i.e fuzzy) way the effect of exogenous disturbances on the output of the linear plant. We deal with two special cases; the first one asumes unrestricted control vector action in R^m . This case represents the closest formulation to that of Willems' approach [8] to geometric problems within the methodology of "almost invariant subspaces". We remark, however, that our generalization does not englobe Willems theory (notice that a fuzzy controlled invariant subspace does not generalize the concept of an almost controlled invariant subspace). The second case we shall treat, assumes fuzzy restrictions on the control space. These two problems could be termed : Fuzzy Disturbance Decoupling Problem with Unrestricted Controls (FDDPUC) and Fuzzy Disturbance Decoupling Problem with Restricted Controls (FDDPRC) respectively.

Fuzzy Disturbance Decoupling Problem with Unrestricted Controls

Consider the linear dynamic system :

$$\dot{\underline{x}}(t) = A \, \underline{x}(t) + B \, \underline{u}(t) + D \, \underline{q}(t) \tag{3.1}$$

$$\underline{y}(t) = C \, \underline{x}(t) \tag{3.2}$$

where the term $\underline{q}(t)$ is an exogenous disturbance in $\mathcal{Q} = R^p$ whose structure is not precisely known except for the fact that only a qualitative statement is at our disposal about the nature of such acting signal (i.e. only a fuzzy description of the signal is available. e.g. "q(t) is a biased unknown signal whose value fluctuates *around* such value" or else, " $\underline{q}(t)$ is a *small* disturbance which oscillates *very rapidly*", etc.). This disturbance in R^p

is thus best modeled by prescribing a , let us say , γ-fuzzification of $\mathcal{Q}$. The control vector $\underline{u}$ in $\mathcal{U} = R^m$ is totally free while the output vector $\underline{y}(t)$ lies in $\mathcal{Y} = R^q$ for all t.

It is required to find (if possible) a state feedback gain F such that the influence of the disturbance signal q(t) is *very small* on the output y(t). (Alternatively, we could say that it is required that the zero input response of the system, i.e. the response for $\underline{u}(t) \equiv 0 \ \forall \ t$, be *very close* to zero or *almost* zero).

Formally, we are given a fuzzy zero in $\mathcal{Y}$ characterized by a m.f. δ (i.e we are given a δ-fuzzification of $\mathcal{Y}$), and it is required to find a feedback gain F such that if $\mathcal{D}_\gamma$ is the fuzzy image of D ,in the state space, of the γ-fuzzification of $\mathcal{Q}$, then :

$$< (A + BF) \,|\, \mathcal{D}_\gamma > \subset_f \mathrm{Ker}_\delta C \tag{3.3}$$

This condition constitutes the fuzzy disturbance decoupling condition for our problem. Because we have not imposed any (fuzzy) restriction on the input signals, this problem has an important conceptual similarity with that treated by Willems in [8]. One of the differences is that Willems' constraints on the output values,due to the disturbance action on the plant, are of the "hard"type, thus forcing the norm of these outputs to become smaller than a preassigned quantity . On the other hand, we impose a"soft"type of constraint on the zero input response allowing it to stay around the zero value so as to be able, at least from a qualitative viewpoint, to asses our control effectiveness,in the desing problem,by a mere subjective evaluation of the disturbance influence on the observed output. The other main difference stems from the fact that our definitions of fuzzy invariant subspaces and fuzzy controllability subspaces do not generalize those of almost invariant and almost controllability subspaces. Our definitions relate to properties (whether qualitative or vaguely known) of elements which do belong to certain subspaces while Willems' essential concepts relate to properties of subspaces with respect ot points of the space which do not belong to the related subspaces (.i.e systems trajectories).

Condition (3.3) signifies, as usual, a pair of mathematical requirements : one on the support subspace and the other on the m.f.'s. These are :

$$< (A + BF) \,|\, \mathcal{D} > \subset \mathrm{Ker}\, C \tag{3.4}$$

and

$$\overset{n-1}{\underset{i=0}{*}} \gamma \circ D^{-1} (A + BF)^{-i} \leqq \delta \circ C \tag{3.5}$$

The solution to the FDDPUC would then be given by the following theorem :

Theorem 3 FDDPUC is solvable if and only if :

$$\mathcal{V}^*_{\psi^*} \supset_f \mathcal{D}_\gamma \tag{3.6}$$

where $\mathcal{V}^*$ is the supremal (A,B)-invariant subspace contained in the Ker C . This subspace is given by the algorithm in (2.28) with $\mathcal{N}$ = Ker C ,while the corresponding m.f. is generated recursively by means of (2.30). According to (3.6) this m.f. must satisfy : ($\mu_{\mathcal{B}} = 1_{\mathcal{B}}$ i.e $\mu(\underline{x}) = 1$ in $\mathcal{B}$ and zero elsewhere)

$$\psi^* \geqq \gamma \tag{3.7}$$

Fuzzy Disturbance Decoupling Problem with Restricted Controls

This case, slightly generalizes the preceeding one by adscribing a μ-fuzzification to the input space $\mathcal{U}$. We now make use of the $(A,B)_\mu$-invariant subspace concept, in order to propose a solution to our problem. In the preceeding problem only (A,B)-invariance had to be invoked due to the"crispness" associated with the input space.

As before, the solution to the problem exists if and only if :

$$\mathcal{V}^*_{\psi^*} \supset_f \mathcal{D}_\gamma \tag{3.8}$$

where V^* is the supremal $(A,B)_\mu$-invariant subspace contained in the fuzzy kernel of C ; $\mathrm{Ker}_\delta C$. ψ^* is given by (2.30)

IV CONCLUSIONS AND SUGGESTIONS FOR FURTHER RESEARCH

In this paper we have given initial steps towards the establishment of a "Fuzzy Geometric Theory of Linear Multivariable Control". We have provided basic definitions that range from fuzzy subspaces and fuzzification of linear vector spaces to Fuzzy Controlled invariant and Fuzzy Controllability subspaces. This framework allows for the adequate treatment, within the fuzzy geometric technique, of wchichever problem has already been formulated and solved by means of Wonham's theory. We can therefore parallel the developments of the Geometric Theory in a fuzzy-theoretic frame.

The results, thus far obtained, generalize those of the existing Geometric Theory of Wonham. This generalization actually represents a needed relaxation of the concepts that researchers have been using since Wonham's early works. This relaxation has a significance, not only from the phylosophical viewpoint, but even from a practical standpoint. This is demonstrated by the closely related line of work initiated by Willems with his "Almost Invariant Subspace theory". Willems methodology, which is rapidly expanding into a new and original approach to a number of classical and modern control problems, could have important inter-relations with the approach we are proposing in this article. Eventhough we did not pay much attention to the similarities and parallelisms that could be exploited and explored with both theoris at hand, we are convinced that an area for further research could be constituted by the bridging of both approaches.

As mentioned before, a wide variety of problems already treated by the conventional Geometric Theory remain unexplored within the Fuzzy Geometric context presented here. Specific examples remain to be worked out and the variety of solutions obtained, analyzed from both a theoretical and practical viewpoint. Problems such as the Decoupling Problem, the Output Regulation Problem, the Disturbance Decoupling with Measurement Feedback and Stability Problem, the Disturbance Decoupling with Measurement Feedback and Pole Placement Problem, the Unknown Input Observer Problem (also known as the Disturbance Decoupled Estimation Problem), etc. deserve attention from a fuzzy geometric viewpoint.

Finally, a natural particularization for the m.f.'s appearing in this work, is the case of "Gaussian Fuzzy m.f.'s " defined over the entire space of subspace. (See [6],[7] for more details). Interesting connections already exist among Fuzzy Sets or variables and random variables and Probability theory . The exploitation of these bridges could shed some light on the recent developments of Stochastic Geometric Theory mentioned by Wonham in [14].

REFERENCES

[1] Wonham, W.M., *Linear Multivariable Control : A Geometric Approach* (2nd. Edition) Springer Verlag. New York. 1979.

[2] Zadeh,L., "Fuzzy Sets," Information and Control., Vol. 8, pp 338-353. 1965.

[3] Zadeh,L., Fu, K.S., Tanaka,K.,and Shimura,M., *Fuzzy Sets and Their Applications to Cognitive Decision Processes*, Academic Press. New York 1975.

[4] Dubois,D., and Prade,H., *Fuzzy Sets and Systems : Theory and Applications*, Mathematics in Science and Engineering. Vol. 144. Academic Press New York. 1980.

[5] Britov,G.S., and Reznik,L.K., "Optimal Control of Linear Fuzzy Systems, " Avtomatika i Telemekhanika, No. 4, April 1981. pp 66-69.

[6] Sira-Ramirez, H.," Evolution of Fuzzy Sets in Linear Dynamic Systems,"

Proceedings of the Internt'l Conference on Cybernetics and Society. Denver, Colorado. October 1979.

[7] Sira-Ramirez,H.," Fuzzy State Estimation in Linear Dynamic Systems," Proceedings of the 19th IEEE Conference on Decision and Control. Vol.1 pp 380-382. Albuquerque, New Mexico. December 1980.

[8] Willems, J.C., " Almost Invariant Subspaces : An Approach to High Gain Feedback Design- Part I : Almost Controlled Invariant Subspaces," IEEE Transactions on Automatic Control. Vol. AC-26, No. 1, pp 235-252. 1981

[9] Willems,J.C., " Almost A(mod $\mathcal{B}$)-Invariant Subspaces," Asterisque, Vol. 75-76, ' Analyse des Systemes' (Bordeaux, 11-16. Sept. 1978) pp. 239-248. 1980.

[10] Willems, J.C., and Commault, Ch., " Disturbance Decoupling by Measurement Feedback with Stability or Pole Placement," SIAM J. on Control and Optimization. Vol. 19, No. 4. July 1981.

[11] Willems,J.C., " Almost Invariant Subspaces: An Approach to High Gain Feedback Design- Part II : Almost Conditionally Invariant Subspaces," Submitted for publication on IEEE Transactions on Automatic Control.

[12] Willems,J.L., and Willems, J.C., " Robust Stabilization of Uncertain Systems," MIT Laboratory for Information and Decision Systems, LIDS-P-1108. July 1981.

[13] Willems, J.C., "Feedforward Control, PID Control Laws, and Almost Invariant Subspaces," MIT Laboratory for Information and Decision Systems LIDS-P-1120. August 1981.

[14] Wonham,W.M.," Geometric State Space Theory in Linear Multivariable Control : A Status Report," Automatica. Vol. 15, No. 1, pp 5-]4. 1979.

APPENDIX

In this appendix we present the basic notation and definitions we use in this article. These definitions were given in [6] with minor modifications.

Definition A1 We denote a fuzzy set A with m.f $\mu_A(\ )$ defined over the universe of discourse U in the n-dimensional euclidean space R^n as :

$$A = \int_U \mu_A(\underline{u}) \,/\, \underline{u} \tag{A.1}$$

A non-fuzzy set will be termed "crisp". Its m.f. has a value 1 over the entire domain of the set. We use $\underline{u} \; \varepsilon_f \; A$ to denote fuzzy membership of $\underline{u}$ to A.

A straightforward application of the Extension Principle [2] allows us to define linear transformations, crisp translations, sums, etc of fuzzy sets in R^n.

Definition A2 We define a crisp translation of a fuzzy set A, in the direction of the vector $\underline{v}$, the fuzzy set given by:

$$A \,\acute{+}\, \{1/\,\underline{v}\,\} = \int_{A\,+\,\underline{v}} \mu_A(\underline{u}-\underline{v}) \,/\, \underline{u} \tag{A.2}$$

where stands for the support of the fuzzy set A (i.e the set of points where the m.f. of A is not zero) " + " denotes the vector sum of the involved sets.

As a generalization of the preceeding definition, consider the fuzzy singleton $\underline{v}$ with m.f. $\mu_B(\underline{v})$, i.e $B \,\acute{=}\, \{\mu_B(\underline{v}) \,/\, \underline{v}\,\}$.

Definition A3 A fuzzy translation of the fuzzy set A in the direction and extent of the fuzzy singleton B , defined above, is defined as a fuzzy set described by:

$$A \,\acute{+}\, \{\mu_B(\underline{v})/\underline{v}\,\} = \int_{A\,+\,\underline{v}} [\ \mu_A(\underline{u}-\underline{v}) \wedge \mu_B(\underline{v})\ ] \,/\, \underline{u} \tag{A.3}$$

where the symbol "$\wedge$ " stands for the infimum function of the two functions values specified to its sides.

As a generalization of the previous definitions, we introduce now the direct sum or vector sum definition of two fuzzy sets. This definition generalizes that of a vector sum of two crisp sets.

Definition A4 Let A be a fuzzy set with m.f. μ_A , and similarly let B be a fuzzy set characterized by μ_B. We define the vector sum of A and B as the fuzzy set specified by:

$$A + B = \int_{\mathcal{A}+\mathcal{B}} \sup_{\underline{v}} [\, \mu_A(\underline{u}-\underline{v}) \wedge \mu_B(\underline{v}) \,] \,/\, \underline{u} \tag{A.4}$$

where $\mathcal{A} + \mathcal{B}$ denotes that the universe of discourse , where the fuzzy sum is to be defined, is the vector sum of the crisp sets that serve as supports for the fuzzy sets A and B respectively. The "sup" operation is necessary to eliminate the possibility of having ill-defined membership values for those elements that can be expressed in a non-unique fashion as sum of elements in $\mathcal{A}$ and $\mathcal{B}$. This supremum operation usually results in a maximization operation.

The above formula (A.4) constitutes a natural "convolution operation" on the m.f's of the fuzzy summands. Naturally, this definition includes the case where A and B are crisp sets. The formula (A.4) is, not surprisingly, reminiscent of that which establishes the probability density function of the sum of two independent random variables. Note also that the roles of $\underline{u}$ and $\underline{v}$ can be interchanged.

Remark It is a basic principle that the result can never exceed the exactitude of the data. Equally true, the sum of two fuzzy sets can not be a crisp set. This fact is established by simple inspection of (A.4).

As a simple application of the extension principle we define now the direct and inverse images of a fuzzy set A in R^n under linear transformations.

Definition A5 Given a non-singular linear transformation P , and a fuzzy set A with m.f. μ_A , we define the direct image under P of the fuzzy set A, the fuzzy set given by :

$$P(A) = \int_{P(\mathcal{A})} \mu_A(P^{-1}\, \underline{u}\,) \,/\, \underline{u} \tag{A.5}$$

Definition A6 The inverse image of a fuzzy set A ,with m.f. $_A$, under the linear map Q , not necessarily invertible, is the fuzzy set described by means of :

$$Q^{-1}(A) = \int_{Q^{-1}(\mathcal{A})} \mu_A(\, Q\, \underline{u}\,) \,/\, \underline{u} \tag{A.6}$$

Acknowledgement The author wishes to thank the enthusiasm and a number of references provided to him by Mr. Daniel Cadorin of the Venezuelan Institute for Scientific Research (I.V.I.C).

CONTROL CONCEPTS FOR FINITE STATE SYSTEMS AND INVERTIBILITY CRITERION*

TOSHIO NOMURA
Department of Mathematics. Southern Illinois University,
Carbondale, Illinois 62901

ABSTRACT: A state space characterization of invertibility of finite state systems is examined in terms of relational calculus and an algorithm which determines the invertibility of systems is obtained, which is shown to be a generalization of the known result in the framework of the linear geometric control. For this purpose two concepts, namely "contractible relation" as an abstract notion of (A,B)-invariant subspace and "product system", are newly developed and utilized intensively.

[1] INTRODUCTION

This paper is part of a series of attempts to enlarge the applicability of control concepts mainly developed for finite dimentional linear systems in the framework of geometric control (Wonham [11]), to more general systems in terms of relational calculus. The major goal is intended toward establishing a control theoretic view of discrete event systems such as production lines of industrial plants, multi robotic hands.

In the literature of Nomura [7], three types of homomorphisms between controlled systems were verified and their systemic significance was investigated in regard to feedback transformations. Then a relationsl congruence was defined as an equivalence kernel of a relational homomorphism between two systems, which was shown to be an abstract generalization of (A,B)-invariant subspace. One feature of this article is to introduce another aspect of (A,B)-invariant subspace, named contractible relation, which refers to the image of the system under a relational homomorphism.

Another feature studied here is to introduce an embedding technique of the originally given system to the "product system", by which various control problems become more naturally formulated.

Let us consider two fundamental problems in the linear control context which concern two distinct characteristics of concept (A,B)-invariant subspace, namely, disturbance decoupling problem and system's invertibility. In a more general setting the characterization of the former problem has been done in terms of the existence of some relational congruence (see [4],[6],[8],[9]). In the present article we will show that the state space characterization of invertibility can be completely formulated by means of the existence of some contractible relation, which gives an abstract algebraic interpretation of the known result in the linear geometric control, "the complete system (A,B,C) with monic B is invertible if and only if $\mathrm{Im}(B) \cap R^* = 0$ where R^* is the largest (A,B)-invariant subspace contained in Ker(c) (see Wonham [11], pp.98).

[2] PRELIMINARIES

Let A, B and C be sets. A relation (or strictly, binary relation) R between A and B is a subset of the Cartesian product of A and B, i.e. $R \subset A \times B$. If Q, $R \subset A \times B$ and $S \subset B \times C$, the composite relation of R and S is defined as $R \circ S = \{(a,c) \in A \times C \mid (a,b) \in R \text{ and } (b,c) \in S \text{ for some } b \in B\}$, and

---

*This research was supported by SERC grant GR/C/23148 when the author was visiting the Department of Mathematics, University of Salford, United Kingdom.

$Q\cap R$ and $Q\cup R$ denote the set-theoretic union and intersection, respectively. The transposed relation R^T of R is defined as $R^T = \{(b,a) \mid (a,b)\in R\}$. Let us define that $L_{A\times B} = \{(a,b) \mid a\in A \text{ and } b\in B\}$. In particular, we will denote that $L_A = L_{A\times A}$. Hence $L_A \circ R \circ L_B = L_{A\times B}$ iff $R\neq\emptyset$. As far as no confusion occurs we may not mention the suffix of L.

The diagonal relation on A is $I_A = \{(a,a) \mid a\in A\}$. In particular, we will denote (a,a) by I_a, therefore, $I_A = \bigcup_{a\in A} I_a$.

The relation $R\subset A\times B$ is called unique if $R^T\circ R\subset I_B$ (that is, $\{b\in B \mid (a,b)\in R\}$ has at most a single element for all a) and is called total if $R\circ R^T \supset I_A$, or equivalently, if $R\circ L_B = L_{A\times B}$ (that is, $\{b\in B\mid (a,b)\in R\}$ is nonempty for all a). Thus a mapping h of A into B is a relation which is unique and total. Further we will say that R is injective (surjective) if R^T is unique (total), or equivalently, if $R\circ R^T\subset I_A$ ($R^T\circ R\supset I_B$).

Let $R\subset A\times A$. It is called a reflexive (or for short, R-) relation if $R\supset I_A$, a symmetric (S-) relation if $R = R^T$, and a transitive (T-) relation if $R\circ R\subset R$.

If R is a RST-relation it is called an equivalence relation which uniquely corresponds to a disjoint partition on A. If R is a ST-relation it uniquely defines a disjoint partition of a subset of A.

Let us call a ST-relation R connected if $(R\cap I_A)\circ L_A\circ (R\cap I_A)\subset R$, that is, if, for any a, $b\in A$, I_a and $I_b\in R$ imply $(a,b)\in R$. An interesting fact is that any subset B of A is realized as a connected ST-relation by the mapping $\nu:\mathcal{P}(A)\to\mathcal{P}(A\times A)$, $B\mapsto\nu(B) = \{(a,b)\in A\times A \mid a,b\in B\}$ where $\mathcal{P}(A)$ denotes the power set of A and the like. Conversely, a connected ST-relation R on A induces a subset of A according to the mapping $\pi:\mathcal{P}(A\times A)\to\mathcal{P}(A)$, $R\mapsto\pi(R) = \{a\in A\mid I_a\in R\}$.

A lattice is a partially ordered set (poset) any two of whose elements, p and q, have a greatest lower bound $p\wedge q$ and a least upper bound $p\vee q$. If every subset of a poset has a greatest lower bound and a least upper bound it is called a complete lattice. In particular, a complete lattice P contains $\vee P = 1p$ and $\wedge P = 0p$.

For example, a set of relations between A and B is a poset according to the order $R < Q$ if $R\subset Q$ as subsets of $A\times B$. By this partial ordering the sets of RST-relations and ST-relations on A become lattices, in particular, complete lattices.

Finite state systems (FSS) which we will be studying are described in a sextuple $A = (\Sigma, Q, \Gamma, \phi, \lambda, q_0)$ denoting, respectively, the input alphabet, the state set, the output alphabet (each of finite order), the state transition function $\phi: \Sigma\times Q\to Q$, the output function $\lambda: Q\to\Gamma$, and the initial state.

Given FSS A we can construct another FSS $\underline{A} = (\underline{\Sigma},Q,\underline{1},\underline{\phi},\underline{\lambda},Iq_0)$, called the product FSS of A, where $\underline{\Sigma} = \Sigma\times\Sigma$, $\underline{Q} = Q\times Q$, $\underline{1} = \{0,1\}$, and

$$\underline{\phi}\colon \underline{\Sigma}\times\underline{Q}\rightarrow\underline{Q}$$
$$(\underline{\sigma},\underline{q}) = ((\sigma_1,\sigma_2),(q_1,q_2)) \mapsto \underline{\phi}(\underline{\sigma},\underline{q}) = (\phi(\sigma_1,q_1),\ \phi(\sigma_2,q_2)),$$

and

$$\underline{\lambda}\colon \underline{Q}\rightarrow\underline{1}$$
$$\underline{q} = (q_1,q_2) \mapsto \underline{\lambda}(q) = \begin{cases} 0 \text{ if } \lambda(q_1) = \lambda(q_2) \\ 1 \text{ otherwise} \end{cases}$$

Let ϕ and $\underline{\phi}$ be state transitions defined above. These mappings naturally induce relations $\Phi\subset Q\times Q$ and $\underline{\Phi}\subset\underline{Q}\times\underline{Q}$, called the successor relations of ϕ and $\underline{\phi}$, respectively, according to: for q, $q'\in Q$, $(q,q')\in\Phi$ if there exists $\sigma\in\Sigma$ satisfying $q' = \phi(\sigma,q)$; and similarly, for $\underline{q},\underline{q}'\in\underline{Q}$, $(\underline{q},\underline{q}')\in\underline{\Phi}$ if there exists $\underline{\sigma}\in\underline{\Sigma}$ such that $\underline{q}' = \underline{\phi}(\underline{\sigma},\underline{q})$; more precisely, if $\underline{q} = (q_1',q_2')$ and $\underline{\sigma} = (\sigma_1,\sigma_2)$ then $(\underline{q},\underline{q}')\in\underline{\Phi}$ iff $q_i' = \phi(\sigma_i,q_i)$, $i = 1,2$.

Let λ and $\underline{\lambda}$ be output functions of A and $\underline{A}$, respectively. We will denote that $N = \lambda\circ\lambda^T\subset Q\times Q$ and $\underline{N} = \underline{\lambda}\circ I_0\circ\underline{\lambda}^T \subset \underline{Q}\times\underline{Q}$ which are a RST-relation on Q and a connected ST-relation on $\underline{Q}$, respectively. More precisely, $(q,q')\in N$ iff $\lambda(q) = \lambda(q')$, and $(\underline{q},\underline{q}')\in\underline{N}$ iff $\underline{\lambda}(\underline{q}) = 0 = \underline{\lambda}(\underline{q}')$, that is, if $\underline{q} = (q_1,q_2)$ and $\underline{q}' = (q_1',q_2')$ then $(\underline{q},\underline{q}')\in\underline{N}$ iff $\lambda(q_1) = \lambda(q_2)$ and $\lambda(q_1') = \lambda(q_2')$, i.e. $(q_1,q_2)\in N$ and $(q_1',q_2')\in N$. Hence $\pi(\underline{N}) = N$ where $\pi\colon P(\underline{Q}\times\underline{Q})\rightarrow P(\underline{Q})$, $\underline{N}\mapsto\pi(\underline{N}) = \{\underline{q}\in\underline{Q} \mid (\underline{q},\underline{q})\in\underline{N}\}$, is the projection onto the first component of $\underline{N}$. Similarly, $\nu(N) = \underline{N}$ where $\nu\colon P(\underline{Q})\rightarrow P(\underline{Q}\times\underline{Q})$, $N\mapsto\nu(N) = \{(\underline{q},\ \underline{q}')\in\underline{Q}\times\underline{Q} \mid \underline{q},\underline{q}'\in N\}$.

As we will see in the section 4, various questions raised about A are nicely resolved in the setting of the product FSS $\underline{A}$.

[3] CONTRACTIBLE RELATION

Let us consider two state transition systems $A_1 = (\Omega,X,\psi)$ and $A_2 = (\Sigma,Q,\phi)$ where $\psi\colon \Omega\times X\rightarrow X$ and $\phi\colon\Sigma\times Q\rightarrow Q$ define the state transitions. By $\Psi\subset X\times X$ and $\Phi\subset Q\times Q$, we will denote the successor relations of ψ and ϕ , respectively.

Each successor relation gives the information concerning where the next state could be by applying some input, rather than that which individual input carries which state to where. In this view each transition system is a relational system consisting of a carrier set $X(Q)$ with binary relation Ψ (Φ).

We call the pair of mappings $f\colon\Omega\rightarrow\Sigma$ and $h\colon X\rightarrow Q$ an algebraic (or, briefly A-) homomorphism if $\phi(f(w),\ h(x)) = h(\psi(w,x))$ for all $w\in\Omega$, $x\in X$, or, the diagram in Fig. 1-A commutes.

We call a mapping $h\colon X \to Q$ a $\cup/\cap/R$-homomorphism (where "/" means "or" and "R" stands for "relational") if $h\circ\Phi \subset \Psi\circ h / h\circ\Phi \supset \Psi\circ h / h\circ\Phi = \Psi\circ h$, respectively, or their diagramatic expression is given in Fig. 1-B,C,D.

One interesting feature here is that if h is a surjective $\cup$-homomorphism, A_2 is a cruder version of A_1, that is, h is a projection, provided inputs of A_1 be rearranged appropriately, and if h is an injective $\cap$-homomorphism, A_1 can be inserted into A_2, that is, h become an inclusion by reorganizing inputs of A_2 properly.

As for the further precise discussion of this matter, in particular in relation to feedback action, the reader should refer to Nomura [7].

If h is a $\cup$-homomorphism the equivalence kernal induced by h in X, i.e. $\mathcal{S} = h\circ h^T \subset X\times X$ has a property

$$\mathcal{S}\circ\Psi \subset \Psi\circ\mathcal{S}$$

that has an exact analogue to (A, B)-invariant subspace in linear systems including relation to feedback, which is called an R-congruence (see [7] for the details).

If $h\colon X\to Q$ is a mapping its image is $h(X)$ which is realized as a connected ST-relation $\mathcal{R} = h^T\circ L_X\circ h \subset Q\times Q$, that is, $(q,q') \in \mathcal{R}$ iff there exist $x, x' \in X$ satisfying $q = h(x)$ and $q' = h(x')$.

LEMMA 1: Suppose $h\colon X\to Q$ is a $\cap$-homomorphism of A_1 into A_2. Then the h-image $\mathcal{R} = h^T\circ L_X\circ h$ satisfies $\mathcal{R} \subset \Phi\circ\mathcal{R}$.

Proof:
$$\begin{aligned}
\Phi\circ\mathcal{R} &= \Phi\circ h^T\circ L_X\circ h && \text{(by definition of } \mathcal{R}\text{)}\\
&\supset h^T\circ h\circ\Phi\circ h^T\circ L_X\circ h && \text{(since } h \text{ is unique, i.e. } h^T\circ h \subset I_Q\text{)}\\
&\supset h^T\circ\Psi\circ h\circ h^T\circ L_X\circ h && \text{(since } h \text{ is a } \cap\text{-homomorphism, i.e. } h\circ\Phi \subset \Psi\circ h\text{)}\\
&\supset h^T\circ\Psi\circ L_X\circ h && \text{(since } h \text{ is total, i.e. } h\circ h^T \supset I_X\text{)}\\
&= h^T\circ L_X\circ h && \text{(since } \Psi \text{ is total, i.e. } \Psi\circ L = L\text{)}\\
&= \mathcal{R} && \text{(by definition of } \mathcal{R}\text{).} \qquad \square
\end{aligned}$$

$$\begin{tikzcd}
\Omega\times X \arrow[r, "\psi"] \arrow[d, "{(f,h)}"'] & X \arrow[d, "h"] \\
\Sigma\times Q \arrow[r, "\Phi"'] & Q
\end{tikzcd}$$

Fig. 1-A

$$\begin{tikzcd}
X \arrow[r, "\Psi"] \arrow[d, "h"'] \arrow[dr, phantom, "\cup"] & X \arrow[d, "h"] \\
Q \arrow[r, "\Phi"'] & Q
\end{tikzcd}$$

Fig. 1-B

$$\begin{tikzcd}
X \arrow[r, "\Psi"] \arrow[d, "h"'] \arrow[dr, phantom, "\cap"] & X \arrow[d, "h"] \\
Q \arrow[r, "\Phi"'] & Q
\end{tikzcd}$$

Fig. 1-C

$$\begin{tikzcd}
X \arrow[r] \arrow[d, "h"'] \arrow[dr, phantom, "\bigcirc"] & X \arrow[d, "h"] \\
Q \arrow[r, "\Phi"'] & Q
\end{tikzcd}$$

Fig. 1-D

This lemma suggests the following definition which gives another facet of abstract (A,B)-invariant subspace. We will consider FSS A and its product FSS $\underline{A}$, and their successor relations Φ and $\underline{\Phi}$, respectively, as defined in section 2:

DEFINITION 1: A connected ST-relation $\mathcal{R} \subset Q \times Q$ ($\underline{\mathcal{R}} \subset \underline{Q} \times \underline{Q}$) is called contractible with respect to ϕ ($\underline{\phi}$) if

$$\mathcal{R} \subset \Phi \circ \mathcal{R} \tag{1}$$

$$(\underline{\mathcal{R}} \subset \underline{\Phi} \circ \underline{\mathcal{R}}) \tag{2}$$

$\square$

Remarks:

(i) The statement (1) is to be read as : for $q, q' \in Q$, if $(q, q') \in \mathcal{R}$ we can select $\sigma \in \Sigma$ satisfying $(\phi(\sigma, q), q') \in \mathcal{R}$. By using the symmetricity of $\mathcal{R}$, similarly, we can choose $\sigma' \in \Sigma$ satisfying $(\phi(\sigma', q'), q) \in \mathcal{R}$. Hence, by transitivity, (1) implies that if $(q, q') \in \mathcal{R}$ then $(\phi(\sigma, q), \phi(\sigma', q') \in \mathcal{R}$ by suitably choosing $\sigma, \sigma' \in \Sigma$, while $q \sim_{\mathcal{R}} \phi(\sigma, q) \sim_{\mathcal{R}} \phi(\sigma', q') \sim_{\mathcal{R}} q'$.

(ii) If we interpret the above case in the level of the state space Q, $\mathcal{R}$ is contractible with respect to ϕ iff, for any q in subset $\pi(\mathcal{R})$ of Q, there exists $\sigma \in \Sigma$ satisfying $\phi(\sigma, q) \in \pi(\mathcal{R})$ where $\pi(\mathcal{R})$ is the projected image of $\mathcal{R}$ onto the first component. Thus the transition leaving any point in $\pi(\mathcal{R})$ can be contracted within $\pi(\mathcal{R})$ by an appropriate choice of input.

(iii) In the statement of (2), since $\mathcal{R}$ is a connected ST-relation of $\underline{Q}$ $\pi(\underline{\mathcal{R}})$ is a subset of $\underline{Q}(=Q \times Q)$. By using a similar interpretation to the above remark (ii), we have : if $q = (q_1, q_2)$ is in $\pi(\underline{\mathcal{R}})$ there exists $\underline{\sigma} = (\sigma_1, \sigma_2) \in \underline{\Sigma}(=\Sigma \times \Sigma)$ satisfying $\underline{\phi}(\underline{\sigma}, \underline{q}) \in \pi(\underline{\mathcal{R}})$, that is $(\phi(\sigma_1, q_1), \phi(\sigma_2, q_2)) \in \pi(\underline{\mathcal{R}})$.

In the rest of this section we will develop an effective procedure of calculating the largest contractible relation with respect to $\underline{\phi}$ contained in connected ST-relation $\underline{\mathcal{N}}$.

LEMMA 2: Let $\underline{\mathcal{R}}$ be a connected ST-relation on $\underline{Q}$.

The following are equivalent:

(i) $\underline{\mathcal{R}} \subset \underline{\Phi} \circ \underline{\mathcal{R}}$

(ii) $\underline{\mathcal{R}} \subset \underline{\mathcal{R}} \circ \underline{\Phi}^T$.

Proof: The equivalence follows by the symmetricity of $\underline{\mathcal{R}}$. $\square$

LEMMA 3: If $\underline{\mathcal{R}}^i$, $i \in I$, are contractible relations with respect to $\underline{\phi}$ then so is $\vee_{i \in I} \underline{\mathcal{R}}^i$.

Remark: Thus the class of contractible relations has the structure of a complete lattice.

Proof: First let us recall the construction; $\vee_{i\in I}\,\underline{R}^i = \cup_{k\geq 0}\, T_k$ where $T_0 = \cup_i \underline{R}^i$ and $T_k = T_{k-1}\circ T_0$.

The proof is by induction on k.

Since $\underline{R}^i \subset \underline{\Phi}\circ\underline{R}^i$, $i\in I$, it follows that $T_0 = \cup_i \underline{R}^i \subset \cup_i \underline{\Phi}\circ\underline{R}^i = \underline{\Phi}\circ\cup_i \underline{R}^i = \underline{\Phi}\circ T_0$.

Suppose that $T_k \subset \underline{\Phi}\circ T_k$ is true. Then $T_k\circ\underline{R}^i \subset \underline{\Phi}\circ T_k\circ\underline{R}^i$, $i\in I$, hence

$$T_{k+1} = T_k\circ T_0 = T_k\circ\cup_i\underline{R}^i = \cup_i T_k\circ\underline{R}^i \subset \cup_i\underline{\Phi}\circ T_k\circ\underline{R}^i = \underline{\Phi}\circ T_k\circ\cup_i\underline{R}^i = \underline{\Phi}\circ T_k\circ T_0 = \underline{\Phi}\circ T_{k+1}.$$

☐

By CR($\underline{\phi}$,$\underline{N}$) we will let us denote the set of contractible relations with respect to $\underline{\phi}$ which is contained in a nonempty connected ST-relation $\underline{N}$ on $\underline{Q}$.

THEOREM 1: There exists a supremum element $\underline{R}^*$ in CR($\underline{\phi}$,$\underline{N}$). Further, $\underline{R}^*$ can be inductively constructed by a nonincreasing sequence of connected ST-relations $\underline{R}_i$ on $\underline{Q}$, defined according to:

$$\underline{R}_0 = \underline{N}$$
$$\underline{R}_{k+1} = (\underline{\Phi}\circ\underline{R}_k)\cap(\underline{R}_k\circ\underline{\Phi}^T)\cap\underline{N}$$
$$\underline{R}^* = \cap_{k\geq 0}\,\underline{R}_k.$$

(See APPENDIX for proof.) ☐

[4] INVERTIBILITY$^{(1),(2),(3),(5),(8),(10)}$

Let $A = (\Sigma, Q, \Gamma, \phi, \lambda, q_0)$ be FSS. We let us denote by Σ^* the free monoid generated over Σ. The state transition function then has a natural extension

$$\phi^* : \Sigma^* \times Q \rightarrow Q$$

which is recursively defined as: (i) $\phi^*(\varepsilon, q) = q$, (ii) $\phi^*(\sigma\sigma', q) = \phi^*(\sigma', \phi^*(\sigma, q))$, where ε is the empty string over Σ^*, and $\sigma, \sigma' \in \Sigma^*$, and $\sigma\sigma'$ means the concatenation of input strings, σ followed by σ'. In particular, by $\phi^*_k(\sigma, q)$ we will denote the k-th state of the transition when $\sigma \in \Sigma^*$ is applied, where the length of σ, $|\sigma|$, is greater than or equal to k.

The input/output function of A is a mapping

$$\rho : \Sigma^* \rightarrow \Gamma^*,\ \sigma \mapsto \rho(\sigma) = (\rho_0(\sigma), \rho_1(\sigma), \ldots, \rho_N(\sigma))$$

where $\rho_k(\sigma) = \lambda(\phi^*_k(\sigma, q_0))$ and $N = |\sigma|$.

DEFINITION 2: FSS A is invertible if the input/output function ρ is injective for input sequences with infinite length, that is, distinct input sequences with infinite length yield different output sequences. □

Before we state the main result of this paper we will introduce some basic assumptions on A in order to avoid various trivialities. Namely, we presume that A is reachable (any state of Q can be reached from q_0 by choosing some input sequence) and reduced (there are no two distinct states which cannot be distinguished in outputs by applying any input sequences), and the mapping $\phi(-,q)\colon \Sigma \to Q$ is injective for all $q \in Q$ (each input works effectively so that exactly one input is associated with each edge in the state transition).

First we collect our assumptions:

$\Phi \subset Q\times Q : \{(q,q') \mid q' = \phi(\sigma,q) \text{ for some } \sigma \in \Sigma\}$,

$\underline{R}^* \subset \underline{Q}\times\underline{Q}$: the supremum element in CR($\underline{\phi}$;$\underline{N}$),

$\pi(\underline{R}^*) \subset \underline{Q} = Q\times Q : \{\underline{q} \mid (\underline{q},\underline{q}) \in \underline{R}^*\}$, the projected image of $\underline{R}^*$,

$I_Q = \{(q,q) \mid q \in Q\}$.

THEOREM 2: FSS A is invertible if and only if

$$\Phi^T\circ\Phi \cap \pi(\underline{R}^*) \subset I_Q \tag{3}$$

Remark: The condition (3) should be read as follows:
for $q_1, q_2 \in Q$, if $\underline{q} = (q_1,q_2) \in \Phi^T\circ\Phi$ there exist $r \in Q$ and $\underline{\sigma} = (\sigma_1,\sigma_2) \in \underline{\Sigma}$ satisfying $q_1 = \phi(\sigma_1,r)$ and $q_2 = \phi(\sigma_2,r)$. If $\underline{q} \in \pi(\underline{R}^*)$ as well, since $\underline{R}^* \in$ CR($\underline{\phi}$;$\underline{N}$), it follows that $\underline{\lambda}(\underline{q}) = 0$, that is $\lambda(q_1) = \lambda(q_2)$. Furthermore, there exists $\underline{\gamma} = (\gamma_1,\gamma_2) \in \underline{\Sigma}$ satisfying $\underline{\phi}(\underline{\gamma},\underline{q}) = (\phi(\gamma_1,q_1), \phi(\gamma_2,q_2)) \in \pi(\underline{R}^*)$, and so $\lambda(\phi(\gamma_1,q_1)) = \lambda(q(\gamma_2,q_2))$, and so on. The condition (3) asserts that the existence of such $\underline{q} = (q_1,q_2)$ has to imply $q_1 = q_2$.

Proof: If A is not invertible there exist at least two distinct input sequences $\sigma_1,\sigma_2 \in \Sigma^*$ of infinite length which yield the same output sequence. Let us define a subset $R \subset Q\times Q$ by : $(q_1,q_2) \in R$ iff $q_i^k = \phi_k^*(\sigma_i,q_0)$, $i = 1,2$, $k \geq 0$. By $\tilde{R}$ we will denote the symmetric, reflexive closure of R, i.e., $\tilde{R} = R \cup R^T \cup I_Q$.

It is easy to see that $\nu(\tilde{R}) = \{(\underline{q},\underline{q}') \in \underline{Q}\times\underline{Q} \mid \underline{q},\ \underline{q}' \in \tilde{R}\}$, where $\underline{Q} = Q\times Q$, is a contractible relation with respect to $\underline{\phi}$ and is contained in $\underline{N}$, that is, $\tilde{R} \in$ CR($\underline{\phi}$;$\underline{N}$).

Let us define N to be the smallest integer satisfying: $(q_1^k,q_2^k) \in I_Q$, $k = 0,1,\ldots,N-1$, and $(q_1^N,q_2^N) \notin I_Q$. Then, although $\underline{q}^N = (q_1^N,q_2^N)$ is an element of $\Phi^T\circ\Phi \cap \tilde{R}$, therefore, of $\Phi^T\circ\Phi \cap \pi(R^*)$ (since $\tilde{R} \subset \pi(\underline{R}^*)$ because of the maximality of $\underline{R}^*$) it follows that $\underline{q}^N$ is not an element of I_Q, which violates the condition (3). (Note that we used the assumption that $\phi(-,q_i^{N-1})\colon \Sigma \to Q$ is injective)

Conversely if (3) does not hold we can find $\underline{q} = (q_1,q_2) \in \underline{Q}$ satisfying

$\underline{q} \in \Phi^T \circ \Phi \cap \pi(\underline{\mathcal{R}}^*)$ and $q_1 \neq q_2$. Let $r \in Q$ be such that $(r,q_i) \in \Phi$, $i = 1, 2$, and let us define $\underline{\sigma} = (\sigma_1,\sigma_2) \in \underline{\Sigma}$ (where $\sigma_1 \neq \sigma_2$ since $q_1 \neq q_2$) such that $q_i = \phi(\sigma_i, r)$, $i = 1, 2$. Further, since $\underline{q} \in \pi(\underline{\mathcal{R}}^*)$ there exists $\underline{\xi} = (\xi_1,\xi_2) \in \underline{\Sigma}^*$ $(=\Sigma^* \times \Sigma^*)$ which satisfies $\underline{\phi}^*(\underline{\xi},\underline{q}) = (\phi^*(\xi_1,q_1), \phi^*(\xi_2,q_2)) \in \pi(\underline{\mathcal{R}}^*)$, and therefore, $\lambda(\phi^*_k(\xi_1,q_1)) = \lambda(\phi^*_k(\xi_2,q_2))$ for all $k \geq 0$.

Let $\mu \in \Sigma^*$ be such that $r = \phi^*(\mu, q_0)$ whose existence is guaranteed by the reachability assumption of $\mathcal{A}$.

Finally, let us define $\beta_i = \mu\sigma_i\xi_i \in \Sigma^*$, $i = 1,2$, by concantenting input sequences defined above. Then $\beta_1 \neq \beta_2$, while $\lambda(\phi^*_k(\beta_1,q_0)) = \lambda(\phi^*_k(\beta_2,q_0))$ for all $k \geq 0$, that is, $\mathcal{A}$ is not invertible. □

[5] EXAMPLE

Let us consider the state transition which is described in Fig. 2 as a directed graph where the state set $Q = \{0,1,2,3,4\}$. As readily checked a connected ST-relation $\mathcal{R} \subset Q \times Q$ which is uniquely defined from subset $G = \{0,1,4\}$ as $\mathcal{R} = \nu(G)$ (therefore, $\pi(\mathcal{R}) = G$), namely $\mathcal{R} = \{(0,0), (0,1), (0,4), (1,0), (1,1), (1,4), (4,0), (4,1), (4,4)\}$ is a contractible relation with respect to ϕ. In fact, for any q in G (in Fig. 2), there is an edge which leads q to some other point in G. (Here the input alphabet is not specified since only the relational structure is of importance.)

The successor relation Φ is expressed in terms of dots (●) in Fig. 3.

On the other hand, a connected ST-relation $\underline{S} \subset \underline{Q \times Q}$ which is constructed from subset $\underline{H}$ of $\underline{Q}$ $(=Q \times Q)$, described by squares (□) in Fig. 3, as $\underline{S} = \nu(\underline{H})$ (therefore, $\pi(\underline{S}) = \underline{H}$), is contractible with respect to $\underline{\phi}$. Actually, for any point $\underline{q}$ in $\underline{H}$, say (0,4), we can find an edge leading to (4,1), that is 0 → 4 and 4 → 1 in Fig. 2 or Fig. 3 (●) which lies in $\underline{H}$ again.

Let us define output function λ: $Q \to \Gamma = \{a,b\}$ according to

λ: $0,1,2 \mapsto a$, $3,4 \mapsto b$.

The subsets $\Phi^T \circ \Phi$ and $\pi(\underline{\mathcal{R}}^*) \subset Q \times Q$ of this FSS $\mathcal{A}$ are obtained by using

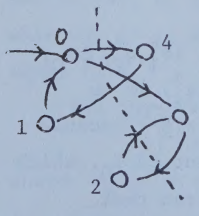


Fig. 2 State Transition

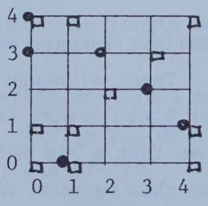


Fig. 3 Φ (●) and $\underline{H}$ (□)

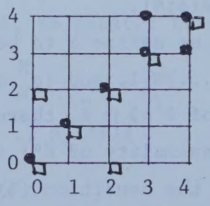


Fig. 4 $\Phi^T \circ \Phi$ (●) and $\pi(\mathcal{R}^*)$ (□)

Fig. 2, Fig. 3 (●), and the algorithm in Theorem 1, and those are described in Fig. 4 in terms of dots (●) and squares (◘), respectively. Since $\Phi^T \circ \Phi \cap \pi(R^*) \subset I_Q$ in this case, A is invertible. □

[6] REMARK

In this article, an algorithm which determines invertibility of FSS is developed, involving two new techniques, contractible relation and product system. A similar approach based on relational calculus will also give constructive characterizations of other control problems such as disturbance decoupling and regulation of FSS.

REFERENCES

(1) BROCKET, R.W. & M.D. MESAROVIC, The Reproducibility of Multivariable Systems, J. Math. Anal. Appl. Vol. 11, 546/563 (1965).

(2) HIRSCHORN, R.M., Invertibility of Control System on Lie Groups. SIAM J. Control & Optimization, Vol. 15, No. 6, 1034/1049 (1977).

(3) HIRSCHORN, R.M., Invertibility of Multivariable Nonlinear Control Systems, IEEE, Vol. AC-24, No. 6, 855/865 (1979).

(4) LIEPA, P.E. & W.M. WONHAM, Feedback Systems in a General Algebraic Setting, IEEE Vol. CAS-25, No. 9, 728/741 (1978).

(5) MASSEY, J.L. & M.K. SAIN, Inverses of Linear Sequential Circuits, IEEE Vol. C-17, 330/334 (1968).

(6) NOMURA, T. & K. FURUTA, Invariant Structures of General Dynamical Systems, SIAM J. Control and Optimization, Vol. 19, No. 1, 154/167 (1981).

(7) NOMURA, T., Homomorphisms and Congruences for Abstract Feedback Systems, to be published in INFORMATION & CONTROL.

(8) NOMURA, T., Invariant Concepts for Finite State Control Systems, The 8th Triennal World Congress of IFAC, Kyoto (1981).

(9) RAMADGE, P.J., & W.M. WONHAM, Algebraic Decomposition of Controlled Sequential Machines, ibid.

(10) WILLSKY, A.S., Invertibility of Groups Systems, Information and Control, Vol. 27, 126/147 (1975).

(11) WONHAM, W.M., Linear Multivariable Control: A Geometric Approach (2nd Edition) Springer-Verlag (1979).

APPENDIX (proof of Theorem 1): It is noticed that $CR(\underline{\phi};\underline{N})$ is nonempty since $I_{\underline{Q}} = \{(Iq,Iq) \in \underline{Q} \times \underline{Q} \mid q \in Q\}$ is its element. Then the existence of the supremum element $\underline{R}^*$ follows immediately from Lemma 3.

To show that $\{\underline{R}_k\}$ is nonincreasing, first notice that $\underline{R}_0 \supset \underline{R}_1$. If $\underline{R}_{k-1} > \underline{R}_k$ then $\underline{R}_k = (\underline{\Phi} \circ \underline{R}_{k-1}) \cap (\underline{R}_{k-1} \circ \underline{\Phi}^T) \cap \underline{N} \supset (\Phi \circ \underline{R}_k) \cap (\underline{R}_k \circ \underline{\Phi}^T) \cap \underline{N} = \underline{R}_{k+1}$.

To show that $\{\underline{R}_k\}$ is a sequence of connected ST-relations, first $\underline{R}_0^T = \underline{N}^T = \underline{N} = \underline{R}_0$ and $\underline{R}_0 \circ \underline{R}_0 = \underline{N} \circ \underline{N} \subset \underline{N} = \underline{R}_0$. Suppose that $\underline{R}_k^T = \underline{R}_k$ and $\underline{R}_k \circ \underline{R}_k \subset \underline{R}_k$. Then

$$\begin{aligned}\underline{R}_{k+1}^T &= [(\underline{\Phi}\circ\underline{R}_k) \cap (\underline{R}_k\circ\underline{\Phi}^T) \cap \underline{N}]^T\\ &= (\underline{R}_k\circ\underline{\Phi}) \cap (\underline{\Phi}^T\circ\underline{R}_k) \cap \underline{N}\\ &= \underline{R}_{k+1}\end{aligned}$$

$$\begin{aligned}\underline{R}_{k+1}\circ\underline{R}_{k+1} &\subset \underline{R}_k\circ[(\underline{\Phi}\circ\underline{R}_k) \cap (\underline{R}_k\circ\underline{\Phi}^T) \cap \underline{N}]\\ &\subset \underline{R}_k\circ\underline{R}_k\circ\underline{\Phi}^T\\ &\subset \underline{R}_k\circ\underline{\Phi}^T\end{aligned}$$

Similarly,

$$\underline{R}_{k+1}\circ\underline{R}_{k+1} \subset \underline{\Phi}\circ\underline{R}_k \quad \text{and} \quad \subset \underline{N}$$

Hence

$$\begin{aligned}\underline{R}_{k+1}\circ\underline{R}_{k+1} &\subset (\underline{\Phi}\circ\underline{R}_k) \cap (\underline{R}_k\circ\underline{\Phi}^T) \cap \underline{N}\\ &= \underline{R}_{k+1}.\end{aligned}$$

As for connectivity of $\{\underline{R}_k\}$, it is obvious that $\underline{R}_0 = \underline{N}$ is connected. Suppose that $\underline{R}_k$ is connected, i.e. $(\underline{R}_k \cap I_Q)\circ L\circ(\underline{R}_k \cap I_Q) \subset \underline{R}_k$. Then, since $(\underline{R}_{k+1} \cap I)\circ(\underline{R}_{k+1} \cap I) = \underline{R}_{k+1} \cap I$, we have:

$$\begin{aligned}(\underline{R}_{k+1} \cap I)\circ L\circ(\underline{R}_{k+1} \cap I) &= (\underline{R}_{k+1} \cap I)\circ(\underline{R}_{k+1} \cap I)\circ L\circ(\underline{R}_{k+1} \cap I)\\ &\subset \underline{R}_{k+1}\circ(\underline{R}_k \cap I)\circ L\circ(\underline{R}_k \cap I) && \text{(since } \underline{R}_{k+1} \subset \underline{R}_k\text{)}\\ &\subset (\underline{\Phi}\circ\underline{R}_k)\circ\underline{R}_k && \text{(by definition of } \underline{R}_{k+1} \text{ and the assumption)}\\ &\subset \underline{\Phi}\circ\underline{R}_k && \text{(by transitivity of } \underline{R}_k\text{)}\end{aligned}$$

Similarly, we have

$$(\underline{R}_{k+1} \cap I)\circ L\circ(\underline{R}_{k+1} \cap I) \subset \underline{R}_k\circ\underline{\Phi}^T \quad \text{and} \quad \subset \underline{N}.$$

Hence $\underline{R}_{k+1}$ is connected.

Thus we proved that $\underline{R}^* = \cap_k\underline{R}_k$ is a connected ST-relation. Furthermore, $\underline{R}^* \in \mathrm{CR}(\underline{\phi};\underline{N})$. In order to show that $\underline{R}^*$ is the largest element in $\mathrm{CR}(\underline{\phi};\underline{N})$, assume that $\underline{R} \in \mathrm{CR}(\underline{\phi};\underline{N})$. Then $\underline{R} \subset \underline{\Phi}\circ\underline{R}$ and $\underline{R} \subset \underline{N}$, hence

$$\underline{R} \subset (\underline{\Phi}\circ\underline{R}) \cap (\underline{R}\circ\underline{\Phi}^T) \cap \underline{N} \qquad \text{(by Lemma 2)}$$

To show that $\underline{R} \subset \underline{R}^*$, first $\underline{R} \subset \underline{N} = \underline{R}_0$. If $\underline{R} \subset \underline{R}_k$ then

$$\begin{aligned}\underline{R} &\subset (\underline{\Phi}\circ\underline{R}) \cap (\underline{R}\circ\underline{\Phi}^T) \cap \underline{N}\\ &\subset (\underline{\Phi}\circ\underline{R}_k) \cap (\underline{R}_k\circ\underline{\Phi}^T) \cap \underline{N}\\ &= \underline{R}_{k+1}\ .\end{aligned}$$

Therefore, $\underline{R} \subset \underline{R}_{k+1}$ for all k, thus $\underline{R} \subset \cap_k\underline{R}_k = \underline{R}^*$. □

RICCATI EQUATIONS AND THE MATRIX EIGENVALUE PROBLEM

G. AMMAR and C. MARTIN
Case Western Reserve University
Cleveland, Ohio

I. INTRODUCTION

The matrix Riccati equation plays a central role in control theory because it appears in the solution of the linear regulator problem. In this context, the correspondence between the equilibria of the Riccati equation and the invariant subspaces of the Hamiltonian matrix is well known. Recent work on the applications of differential geometry to control theory indicates that Lie theory provides a natural setting for the study of Riccati equations. We show here that the classical algorithms for finding the invariant subspaces of a matrix are intimately related to the generalized Riccati equations. To illustrate our approach we will first consider the power method, which is the simplest of these algorithms.

Let A be a real $n \times n$ matrix. The power method is classically defined as the repeated action of A on the unit sphere $S^{n-1} \subseteq \mathbb{R}^n$: starting at $x_0 \in S^{n-1}$, a sequence $\{x_i\}$ is generated according to the formula $x_i = Ax_{i-1}/\|Ax_{i-1}\|$ $(i = 1,2,\ldots)$. It is well known (see, for example [5]) that if A has a simple eigenvalue λ with modulus exceeding that of all other eigenvalues (i.e., if it has a positive dominant eigenvalue), then the sequence $\{x_i\}$ will converge to the unit eigenvector v (or $-v$) corresponding to λ for almost all choices of x_0. More precisely, convergence to v or $-v$ is guaranteed provided x_0 is not contained in an invariant subspace of A which excludes v.

The concern here, of course, is the calculation of an invariant subspace of the matrix A. The power iterates x_i defined above are actually representatives of the subspaces $X_i = \text{span}\{x_i\}$ $(i = 1,2,\ldots)$. The set of all one-dimensional subspaces of $\mathbb{R}^n$ is an (n-1)-dimensional real manifold called real projective space and denoted by $\mathbb{RP}^{n-1}$. This is the natural space on which to study the power iteration.

A set of charts for $\mathbb{RP}^{n-1}$ is defined as follows. Given $U \in \mathbb{RP}^{n-1}$, choose any (n-1)-dimensional subspace W or $\mathbb{R}^n$ such that $U \cap W = \{0\}$, and let $L(U,W)$ denote the set of linear maps from U to W. Note that $L(U,W)$ can be identified with $\mathbb{R}^{n-1}$. For each $K \in L(U,W)$ let $U_k \in \mathbb{RP}^{n-1}$ be given by

$$U_K = \{u + Ku : u \in U\} \subseteq \mathbb{R}^n \quad ,$$

and define $\Gamma(W)$ to be equal to

$$\{U_K : K \in L(U,W)\} \subseteq \mathbb{RP}^{n-1} \quad .$$

It is easily verified that $U_{K_1} = U_{K_2}$ implies $K_1 = K_2$, so that the correspondence between $L(U,W)$ and $\Gamma(W)$ is bijective. This correspondence

$\phi_W : (W) \to L(U,W)$ is, in fact, an analytic diffeomorphism. In this way, the direct sum decomposition $\mathbb{R}^n = U \oplus W$ determined a coordinate neighborhood $\Gamma(W)$ of U and a chart map $\phi_W : \Gamma(W) \to \mathbb{R}^{n-1}$. A pair $(\Gamma(W), \phi_W)$ defined in this manner will be called a *canonical chart* for $\mathbb{RP}^{n-1}$.

Note that any $U_K \in \Gamma(W)$ satisfies $U_K \cap W = \{0\}$. Conversely, if $V \in \mathbb{RP}^{n-1}$ is such that $V \cap W = \{0\}$, then any nonzero $v_o \in V$ can be uniquely written as $v_o = u_o + w_o$, where $u_o \in U$, $u_o \neq 0$, and $w_o \in W$. The requirement $Ku_o = w_o$ then defines a linear map $K : U \to W$ such that $U_K = V$. The coordinate neighborhood $\Gamma(W)$ can therefore be characterized as the set of one-dimensional subspaces complementary to W. Those subspaces not included in $\Gamma(W)$ (i.e., the one-dimensional subspaces of W) form a submanifold of $\mathbb{RP}^{n-1}$ isomorphic to $\mathbb{RP}^{n-2}$; consequently, $\Gamma(W)$ is open and dense in $\mathbb{RP}^{n-1}$. Each canonical chart can then be viewed as a dense embedding of $L(U,W)$ into $\mathbb{RP}^{n-1}$.

Now consider the power method as an iteration on $\mathbb{RP}^{n-1}$. The action of an $n \times n$ matrix A on $X_o = \text{span}\{x_o\}$ is given by $A \cdot X_o = \text{span}\{Ax_o\}$; this definition is clearly independent of the representative vector x_o of X_o. The power iterates X_i on $\mathbb{RP}^{n-1}$ are then defined by the successive action of A on X_o: $X_i = A \cdot X_{i-1}$ $(i = 1,2,\ldots)$.

The Riccati equation arises when the power iteration on the canonical charts for $\mathbb{RP}^{n-1}$ is considered. Let

$$e_1 = \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \quad e_2 = \begin{pmatrix} 0 \\ 1 \\ \vdots \\ 0 \end{pmatrix}, \quad \ldots \quad e_n = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 1 \end{pmatrix}$$

be the standard basis for $\mathbb{R}^n$, and decompose $\mathbb{R}^n$ as $U \oplus W$, where $U = \text{span}\{e_2, e_3, \ldots, e_n\}$. Any element of $\mathbb{R}^n$ can then be expressed as a partitioned vector $\begin{pmatrix} u \\ w \end{pmatrix}$, where $u \in \mathbb{R}$ and $w \in \mathbb{R}^{n-1}$. Similarly, an $n \times n$ matrix A can be written in partitioned form as

$$\begin{pmatrix} A_1 & A_2 \\ A_3 & A_4 \end{pmatrix},$$

where A_1 is scalar and A_4 is $(n-1) \times (n-1)$. Now let $K \in L(U,W)$ and consider the action of A on $U_K \in \Gamma(W)$:

$$A \cdot U_K = \left\{ \begin{pmatrix} A_1 & A_2 \\ A_3 & A_4 \end{pmatrix} \begin{pmatrix} u \\ Ku \end{pmatrix} : u \in \mathbb{R} \right\}$$

$$= \left\{ \begin{pmatrix} (A_1 + A_2K)u \\ (A_3 + A_2K)u \end{pmatrix} : u \in \mathbb{R} \right\}$$

$$= \left\{ \begin{pmatrix} u \\ (A_3 + A_4K)/(A_1 + A_2K)u \end{pmatrix} : u \in \mathbb{R} \right\} .$$

Hence, $A \cdot U_K = U_{(A_3+A_4K/A_1+A_2K)}$ provided the scalar $A_1 + A_2$ is nonzero. If, on the other hand, $A_1 + A_2K = 0$, then $A \cdot U_K \subseteq W$, which implies $A \cdot U_K \notin \Gamma(W)$. Thus, the power method on the coordinate neighborhood $\Gamma(W)$ of U is equivalent to the iteration on $L(U,W)$ (the set of (n-1) by 1 matrices) given by $K_{i+1} = A_3+A_4K_i/A_1+A_2K_i$, where K_o corresponds to the initial subspace $U_{K_o} \in \Gamma(W)$. By rewriting this as a difference equation, we obtain

$$K_{i+1} - K_i = (A_3 + A_4K_i - K_iA_1 - K_iA_2K_i)/(A_1 + A_2K_i) \quad . \tag{1}$$

The power iterates K_i can therefore be identified with a generalized Euler approximation (with variable step size $1/A_1 + A_2K_i$) to the solution $K(t)$ of the differential Riccati equation

$$\dot{K}(t) = A_3 + A_4K(t) = K(t)A_1 - K(t)A_2K(t), K(0) = K_o \tag{2}$$

on $L(U,W) = \mathbb{R}^{n-1}$. In this way, the convergence properties of the power iteration are directly related to the equilibrium points of (2) and the stability properties of these equilibria.

A vector field on $\mathbb{RP}^{n-1}$ that is represented in local coordinates by an equation of the form (2) is called a *Riccati vector field*. In [4] it is indicated that every Riccati vector field (2) on $\mathbb{RP}^{n-1}$ is the local representation of the infinitesimal action $\dot{X} = A \cdot X$ of A on $\mathbb{RP}^{n-1}$. The manifold $\mathbb{RP}^{n-1}$ can be realized as a homogeneous space of the Lie group $G\ell(\mathbb{R}^n)$, and in this way the power method can be considered as the successive action of $G\ell(\mathbb{R}^n)$ on one of its homogeneous spaces. Therefore the theory of Lie groups provides a natural setting for the study of the power method.

These ideas generalize directly to the more useful algorithms for finding the invariant subspaces of a matrix. We will consider two of these algorithms here: a generalized power iteration (which is equivalent to the classical treppen iteration), and the QR algorithm.

II. THE POWER METHOD ON THE GRASSMANN MANIFOLD

The power method extends quite naturally to an iteration on subspaces of dimension more than one. Let V be an n-dimensional vector space over a field $\mathbb{F}$, where $\mathbb{F}$ is either $\mathbb{R}$ or $\mathbb{C}$, and denote by $G^k(V)$ the *Grassmann manifold* of k-dimensional subspaces of V. An analytic manifold structure for $G^k(V)$ is given as follows. For an invertible linear transformation $A \in G\ell(V)$ and a k-space $U \in G^k(V)$, we define $A \cdot U$ to be equal to

$$\{Au : u \in U\} \quad .$$

Since A is invertible, $A \cdot U \in G^k(V)$. Therefore, the mapping $(A,U) \to A \cdot U$ defines an action of the general linear group $G\ell(V)$ on $G^k(V)$. This action is clearly transitive. Moreover, the isotropy subgroup H of $G\ell(V)$ at U, defined by

$$H = \{A \in G\ell(V) : A \cdot U = U\} \quad ,$$

is a closed subgroup of $G\ell(V)$ of codimension $k \cdot (n-k)$. In this way, the manifold $G^k(V)$ can be identified with the homogeneous space $G\ell(V)/H$.

Now fix $U \in G^k(V)$ and let W be an (n-1)-dimensional subspace of V that is complementary to U. By choosing bases for U and W we can express any element of $V = V \oplus W$ as a partitioned vector $\begin{pmatrix} u \\ w \end{pmatrix}$, where $u \in \mathbb{F}^k$ and $w \in \mathbb{F}^{n-k}$. Similarly, we will represent a linear operator A on V as a partitioned matrix

$$\begin{pmatrix} A_1 & A_2 \\ A_3 & A_4 \end{pmatrix} \quad ,$$

where A_1 and A_4 are square matrices of order k and $n-k$, respectively. For each linear map $K \in L(U,W)$, define $U_K \in G^k(V)$ by

$$U_K = \left\{ \begin{pmatrix} u \\ Ku \end{pmatrix} : u \in \mathbb{F}^k \right\} \quad .$$

Let $\psi : L(U,W) = \mathbb{F}^{k\cdot(n-k)} \to G^k(V)$ be given by $\psi(K) = U_K$, and denote the image of ψ by $\Gamma(W)$.

Proposition. The map ψ is injective. Furthermore, its image $\Gamma(W)$ is open and dense in $G^k(V)$.

Proof. Note that for all $K \in L(U,W)$, $U_K \cap W = \{0\}$. Conversely, let $Z = \text{span}\{z_1, z_2, \ldots, z_k\} \in G^k(V)$ be such that $Z \cap W = \{0\}$. Then each basis vector z_i corresponds to unique vectors $u_i \in \mathbb{F}^k$ and $w_i \in \mathbb{F}^{n-k}$ such that $z_i = \begin{pmatrix} u_i \\ w_i \end{pmatrix}$ and, moreover, $u_i \neq 0$ $(i = 1,2,\ldots,k)$. The relations

$$Ku_i = w_i \qquad (i = 1,2,\ldots,k)$$

then determine a unique linear map $K : U \to W$. Note that, by construction, $\psi(K) = Z$. The map ψ is therefore one-to-one. This argument also shows that $\Gamma(W)$ can be identified as the set of k-dimensional subspaces of V which intersect W trivially. If $Z \in G^k(V)$ is such that $Z \cap W \neq \{0\}$, then any basis of Z must satisfy a nontrivial algebraic relation. The complement of $\Gamma(W)$ is therefore a closed submanifold of $G^k(V)$ with codimension one or more. This proves the proposition.

In [2] it is shown that $\psi : L(U,W) \to \Gamma(W)$ is a diffeomorphism. We can now define the canonical charts for $G^k(V)$ exactly as those for $\mathbb{R}P^{n-1}$ were defined: $\Gamma(W)$ is the coordinate neighborhood corresponding to the decomposition $V = V \oplus W$ and the coordinates of $U_K \in \Gamma(W)$ are the entries of the matrix $K \in L(U,W) = \mathbb{F}^{k\cdot(n-k)}$.

Consider now the action of $A \in G\ell(V)$ on $U_K \in \Gamma(W) \subseteq G^k(V)$:

$$A \ U_K = \left\{ \begin{pmatrix} A_1 & A_2 \\ A_3 & A_4 \end{pmatrix} \begin{pmatrix} u \\ Ku \end{pmatrix} : u \in \mathbb{F}^k \right\}$$

$$= \left\{ \begin{pmatrix} (A_1+A_2K)u \\ (A_3+A_4K)u \end{pmatrix} : u \in \mathbb{F}^k \right\}$$

$$= \left\{ \begin{pmatrix} u \\ (A_3+A_4K)(A_1+A_2K)^{-1}u \end{pmatrix} : u \in \mathbb{F}^k \right\}$$

$$= U_{((A_3+A_4K)(A_1+A_2K)^{-1})} \quad .$$

The action of $G\ell(V)$ on $G^k(V)$ is therefore given in the local coordinates on $\Gamma(W)$ as the generalized linear fractional transformation $K \to (A_3 + A_4K)(A_1 + A_2K)^{-1}$, provided the indicated inverse exists. Thus, the power iterates of A starting at $U_{K_0} \in \Gamma(W)$ are given by the difference equations

$$K_{i+1} - K_i = (A_3 + A_4K_i - K_iA_1 - K_iA_2K_i)(A_1 + A_2K_i)^{-1}$$

on $L(U,W) = \mathbb{F}^{k\cdot(n-k)}$.

This discretization arises in control theory in the discrete-time linear regulator problem. (See [3] for a more complete treatment of the discrete-time linear regulator.) Given a discrete linear system,

$$x_i = Ax_{i-1} + Bu_{i-1} \qquad (i = 1,2,\ldots,N), \quad x_0 \text{ given}$$

where the state vectors $x_i \in \mathbb{R}^k$ and the controls $u_i \in \mathbb{R}^{n-k}$, a sequence $\{u_i\}$ of controls is to be chosen to minimize the quadratic cost functional

$$\frac{1}{2} x_N' S x_N + \frac{1}{2} \sum_{i=0}^{N-1} (x_i' Q x_i + u_i' R u_i) \quad ,$$

where Q and S are symmetric, positive semidefinite matrices; R is a symmetric, positive definite matrix; and x' denotes the transpose of x. The optimal sequence of controls $\{u_i^*\}$, found by using the technqiue of dynamic programming, is given in closed-loop feedback form as

$$u_i^* = (R + B'K_{i+1}B)^{-1} B'K_{i+1}Ax_i \quad ,$$

where the matrices K_i satisfy the discrete Riccati equation

$$K_i = A'K_{i+1}A + Q - (B'K_{i+1}A)'(R+B'K_{i+1}B)^{-1}B'K_{i+1}A \quad ,$$

$$K_N = S \quad .$$

The calculus of variations can also be used to find the optimal control sequence. Using this approach, the solution is given by

$$u_i^* = -R^{-1}R'P_{i+1}^* \quad ,$$

where the partitioned vector $\begin{pmatrix} x_i^* \\ p_i^* \end{pmatrix}$ satisfies the linear Hamiltonian system

$$\begin{pmatrix} x_i \\ p_i \end{pmatrix} = \begin{pmatrix} A^{-1} & A^{-1}BR^{-1}B' \\ QA^{-1} & A + QA^{-1}BR^{-1}B' \end{pmatrix} \begin{pmatrix} x_{i+1} \\ p_{i+1} \end{pmatrix} \quad .$$

These two solutions for u_i^* are equal, of course, and in this way solutions of the discrete Riccati equations can be identified with the power iterates of the associated Hamiltonian matrix.

The power method has been studied as a recursion on the set of k-dimensional subspaces of V in [1] and [5]. The convergence criteria derived in these two papers directly involves the structure of the set $S_A(k)$ of all A-invariant subspaces $U \in G^k(V)$. Given $A \in G\ell(V)$, a subspace $U \in S_A(k)$ is called *dominant* if the eigenvalues of the restriction $A_1 = A|_U$ have modulus greater than or equal to that of all other eigenvalues of A. The following is a restatement of Theorem 4 in [5].

Theorem. Suppose $A \in G\ell(V)$ has a unqiue k-dimensional dominant subspace $U \in S_A(k)$, and further assume that U has an A-invariant complementary subspace $W \in S_A(n-k)$. Then the power iterates of A starting at

$X_0 \in G^k(V)$ will converge linearly to U if and only if $X_0 \cap W = \{0\}$, or equivalently, $X_0 \in \Gamma(W)$.

Let A have eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_n$, repeated according to multiplicity and ordered so that $|\lambda_1| \geq |\lambda_2| \geq \ldots \geq |\lambda_n|$. If $|\lambda_k| > |\lambda_{k+1}|$, the hypotheses of the theorem are satisfied by taking U and W to be the generalized eigenspaces corresponding to $\lambda_1, \ldots, \lambda_k$ and $\lambda_{k+1}, \ldots, \lambda_m$, respectively. Convergence in this case is therefore guaranteed for all starting places X_0 in an open, dense subset of $G^k(V)$. If, on the other hand, $|\lambda_k| = |\lambda_{k+1}|$ the situation is more complicated; the unique dominant subspace $U \in S_A(k)$ may not have an invariant complement, or there may not be such a unique subspace U. The topology of $S_A(k)$ is studied in [6].

Lie groups arise naturally in the study of Riccati equations on $G^k(V)$. The vector field (5) is the local representation of the globally defined infinitesimal action of A on $G^k(V)$; $\dot{X} = A \cdot X$. Solutions to (5) therefore correspond to the global action of the one-parameter subgroup e^{At} on $G^k(V)$. The power iterates are approximations to this global flow.

Lie theory is also useful in studying the stability properties of the equilibria of (5). Since $G^k(V)$ is a homogeneous space of $G\ell(V)$, the tangent space at a point of $G^k(V)$ is isomorphic to the quotient $L(V,V)/\mathbb{H}$, where the set of linear maps $L(V,V)$ is the Lie algebra of $G\ell(V)$ and $\mathbb{H}$ is the Lie algebra of the isotropy subgroup H of $G\ell(V)$. The linearization of the vector field $\dot{X} = AX$ at an equilibrium point $U \in G^k(V)$ is then given by the linear map

$$\begin{aligned} \mathrm{Ad}A : L(V,V)/\mathbb{H} &\to L(V,V)/\mathbb{H} \\ B &\to [A,B] \quad . \end{aligned}$$

The eigenvalues of this linear map determine the behavior of flows generated by (5) in a neighborhood of U.

Specifically, let $U \in G^k(V)$ be an A-invariant subspace and choose any complementary subspace W. Then any element of the isotropy subgroup H of $G\ell(V)$ is a block upper triangular matrix with respect to the decomposition $V = U \oplus W$. Then $L(V,V)/\mathbb{H}$ can be identified as the set of all $B \in L(V,V)$ of the form

$$B = \begin{pmatrix} 0 & 0 \\ B_3 & 0 \end{pmatrix} .$$

The linearization of A at U can now be calculated. (Note that $A_3 = 0$ since $A \cdot U \subseteq U$.)

$$[A,B] = AB - BA$$

$$= \begin{pmatrix} A_1 & A_2 \\ 0 & A_4 \end{pmatrix} \begin{pmatrix} 0 & 0 \\ B_3 & 0 \end{pmatrix} - \begin{pmatrix} 0 & 0 \\ B_3 & 0 \end{pmatrix} \begin{pmatrix} A_1 & A_2 \\ 0 & A_4 \end{pmatrix}$$

$$= \begin{pmatrix} A_2B_3 & 0 \\ A_4B_3 - B_3A_1 & -B_3A_2 \end{pmatrix}$$

Thus, the linearization of A is given by

$$[A,B] = A_4B_3 - B_3A_1 \quad .$$

III. THE QR ALGORITHM

The QR algorithm is the preferred way to find the invariant subspaces of a matrix. Starting with $A = A_o \in G\ell(\mathbb{R}^n)$, a sequence of matrices A_i is generated by

$$A_i = R_{i-1}Q_{i-1} \qquad (i = 1,2,\ldots) \quad ,$$

where $A_{i-1} = Q_{i-1}R_{i-1}$ is the (unique) factorization of A_{i-1} into an orthogonal matrix Q_{i-1} and an upper triangular matrix R_{i-1}.

In [5], it is shown that the QR algorithm is equivalent to a nested sequence of power iterations as follows. For each integer m, let P_m be the orthogonal matrix $P_m = Q_1Q_2 \ldots Q_m$ and let S_m be the upper triangular matrix $S_m = R_mR_{m-1} \ldots R_1$. Note that $P_{m-1}A_m = AP_{m-1}$. Hence, $P_mS_m = P_{m-1}Q_mR_mS_{m-1} = P_{m-1}A_mS_{m-1} = AP_{m-1}S_{m-1}$. Since each R_i is nonsingular, this implies

$$P_mR_m = A \cdot P_{m-1} \quad .$$

Thus, for each $q = 1,2,\ldots,n$, the span of the first q columns of P_m is equal to the m-th power iterate of A starting at the subspace spanned by the first q columns of the identity matrix. In this way, the QR algorithm can be viewed as the simultaneous execution of n power iterations; one on $G^k(V)$ for each value of k. Questions regarding the convergence of the QR algorithm can therefore be answered in terms of the power method.

Let $\sigma = (\sigma_1,\sigma_2,\ldots,\sigma_p)$ be an ordered partition of the integer n, and define the *flag manifold* $\mathfrak{F}(V;\sigma)$ of V corresponding to σ as the set of all p-tuples $(V_1,V_2,\ldots,V_p) \in G^{\sigma_1}(V) \times G^{\sigma_1+\sigma_2}(V) \times \ldots \times G^n(V)$ such that $V_1 \subseteq V_2 \subseteq \ldots \subseteq V_p$. The general linear group $G\ell(V)$ acts transitively on $\mathfrak{F}(V;\sigma)$, and with a closed isotropy subgroup, so $\mathfrak{F}(V;\sigma)$ is a homogeneous space of the Lie group $G\ell(V)$. As indicated above, the

QR algorithm is equivalent to the power iteration on $\mathfrak{F}(V;(1,1,\ldots,1))$ starting at the n-tuple

$$(\mathrm{span}\{e_1\},\ \mathrm{span}\{e_1,e_2\},\ \mathrm{span}\{e_1,e_2,e_3\},\ldots,V) \quad .$$

The tools of Lie theory can therefore be used to study the QR algorithm as a power iteration on the flag manifold.

REFERENCES

1. Buurema, Hendrik Jan, "A Geometric Proof of Convergence for the QR Method," Ph.D. dissertation, Rijksuniversiteit Te Groningen, 1970.

2. Doolin, B. F., and C. F. Martin, "Global Differential Geometry: An Introduction for Control Engineers," NASA Reference Publication 1091, 1982.

3. Dorato, Peter and Alexander H. Levis, "Optimal Linear Regulators: The Discrete-Time Case," IEEE Trans. Aut. Cont., Vol. AC-16, No. 6, Dec. 1971, pp. 613-620.

4. Hermann, Robert, Cartanian Geometry, Nonlinear Waves and Control Theory, Part A, Math Sci Press, Brookline, MA., 1979.

5. Parlett, B. N., and W. G. Poole, Jr., "A Geometric Theory for the QR, LU and Power Iterations," Siam J. Numerical Analysis, Vol. 10, No. 2, April 1973.

6. Shayman, Mark A., "Varieties of Invariant Subspaces and the Algebraic Riccati Equation," Ph.D. dissertation, Harvard University, 1980.

STRUCTURAL STABILIZATION OF UNCERTAIN SYSTEMS: NECESSITY OF THE MATCHING CONDITION[†]

Ian R. Petersen
Department of Electrical Engineering
University of Rochester
Rochester, New York 14627

Abstract: This paper investigates one aspect of the problem of stabilizing an uncertain linear system. That is, the systems under consideration contain uncertain parameters which are unknown but bounded. The question arises as to whether such a system can be stabilized via feedback control. In some of the previous papers in this area, the system is assumed to satisfy a so-called "matching-condition"; this type of assumption is used to assure that the uncertain system can be stabilized. It is, however, known that the matching condition is not a necessary condition for stabilizability. This paper introduces a strengthened notion of stabilizability referred to as *structural stabilizability via a nominally determined quadratic Lyapunov function*. It is then shown that if a system is to have this stronger property, the matching condition must necessarily be satisfied.

I. SYSTEM AND INTRODUCTION

This paper is concerned with a stabilization problem for an uncertain linear dynamical system described by

$$\dot{x}(t) = [A_0 + \Delta A(r(t))]x(t) + B_0 u(t); \qquad (\Sigma)$$
$$r(t) \varepsilon \mathcal{R}$$

where $x(t) \varepsilon R^n$ is the *state*, $u(t) \varepsilon R^m$ is the *control*, $r(t) \varepsilon R^p$ is the *vector of uncertain parameters* and $\mathcal{R} \subset R^p$ is a compact *uncertainty bounding set*. It is assumed that the matrix function $\Delta A(\cdot)$ is continuous and that $\Delta A(\tilde{r}) = 0$ for some $\tilde{r} \varepsilon \mathcal{R}$. The function $r(\cdot)$ is restricted to be a Lebesgue measurable function such that $r(t) \varepsilon \mathcal{R}$ for all $t \geq 0$ and it is furthermore assumed that $m < n$ and rank $B_0 = m$. Associated with (Σ) above is a known system obtained when $\Delta A(r) \equiv 0$. This system, described by the state equation

$$\dot{x}(t) = A_0 x(t) + B_0 u(t) \qquad \mathrm{NOM}(\Sigma)$$

is henceforth referred to as the *nominal system*, NOM(Σ).

When dealing with systems of the form (Σ), it is of interest to know whether asymptotic stability can be guaranteed via the application of a feedback control. That is, can a feedback control be found such that the closed loop system is asymptotically stable for any admissible uncertainty? In [1], NOM(Σ) is assumed to be stabilizable and it is shown that satisfaction of a so-called "matching condition" is sufficient for stabilizability. More precisely, (Σ) is said to satisfy the *matching condition* if there exists a continuous matrix function $D(\cdot): R^p \rightarrow R^{m \times n}$ such that

$$\Delta A(r) = B_0 D(r) \qquad (1.1)$$

for all $r \varepsilon R^p$. It should be noted that the matching condition above is only sufficient for stabilizability and not necessary. That is, there

---

[†] This work was supported by the National Science Foundation under Grant ECS-8108804.

exist systems which fail to satisfy the matching condition and yet are stabilizable. Examples of such systems can be found in [2]-[5]. This paper introduces a stronger notion of stabilizability for which the matching condition is shown to be necessary.

In order to relate this work to existing results, it is important to point out one salient feature of [1]; that is, if (Σ) satisfies the matching condition, then stabilization can be achieved given any uncertainty bounding set $\mathcal{R}$ which is arbitrarily large but finite. The ability to tolerate arbitrarily large perturbations motivates our definition of structural stabilizability. This concept will be formally defined in the next section.

In [1]-[5], a quadratic Lyapunov function of the form $V(x) = x'Px$ is used to establish the stability of the closed loop system. Furthermore, the positive-definite matrix P is also used in the construction of the desired stabilizing feedback control law. The procedure given in [1] for the construction of the matrix P is as follows:

(i) Find an m×n matrix K such that $\bar{A}_0 \overset{\Delta}{=} A_0 + B_0K$ is a stability matrix.

(ii) Choose any positive-definite matrix Q and solve the Lyapunov equation

$$\bar{A}_0'P + P\bar{A}_0 = -Q$$

to obtain the matrix P. Hence, we see that the Lyapunov function obtained by (i) and (ii) is solely determined by the system NOM(Σ). In this paper, we consider the problem of characterizing the class of systems (Σ) for which this "nominally determined" Lyapunov function will work; that is, for what class of systems (Σ) can one use a Lyapunov function generated from NOM(Σ)? Uncertain systems having this property have the advantage that a suitable Lyapunov function is straightforward to find. In the sequel, systems having this property are formally said to be structurally stabilizable via a nominally determined quadratic Lyapunov function. The main result of this paper, Theorem 3.1, can now be paraphrased in a rather compact manner: Namely, (Σ) is structurally stabilizable via a nominally determined quadratic Lyapunov function if and only if the matching condition is satisfied and NOM(Σ) is stabilizable. In contrast to the earlier work dealing only with sufficiency of the matching condition, this paper establishes the fact that the matching condition is also necessary.

II. DEFINITIONS AND NOTATION

The uncertain system (Σ) is said to be quadratically stabilizable if there exists a continuous feedback control function $p(\cdot): R^n \to R^m$ and an n×n positive-definite matrix P leading to the satisfaction of the following condition: The closed loop system

$$\dot{x}(t) = [A_0 + \Delta A(r(t))]x(t) + B_0 p(x(t)),$$
$$r(t) \varepsilon \mathcal{R}$$

with Lyapunov function

$$V(x) = x'Px,$$

is uniformly asymptotically stable in the so-called guaranteed sense. That is, there exists a constant $\beta > 0$ such that for any admissible vector uncertainty function $r(\cdot)$, the Lyapunov derivative admits the bound

$$\begin{aligned} -\beta\|x\|^2 \geq L(x,t) &\overset{\Delta}{=} [\nabla V(x)]'[A_0x + \Delta A(r(t))x + B_0p(x)] \\ &= 2x'P[A_0 + \Delta A(r(t))]x + 2x'PB_0p(x) \end{aligned}$$

for all pairs $(x,t) \varepsilon R^{n+1}$; see also [5].

If P is an n×n positive-definite symmetric matrix such that the above condition is satisfied then the uncertain system (Σ) is said to be *quadratically stabilizable with Lyapunov function* $V(x) = x'Px$.

The uncertain system (Σ) is said to have *linear uncertainty structure* if the matrix function $\Delta A(\cdot)$ is linear. That is, $\Delta A(r)$ can be written in the form

$$\Delta A(r) = \sum_{i=1}^{p} A_i r_i$$

where $A_1, A_2, \ldots, A_p$ are constant n×n matrices and r_i is the i^{th} component of the vector r.

A. Systems with Linear Uncertainty Structure

The definitions presented in this sub-section will apply only to systems with linear uncertainty structure. These definitions will be generalized in the next sub-section.

For the linear case, the uncertain system (Σ) is said to be *structurally stabilizable* if for each compact set $\mathcal{R} \subset R^p$, the system (Σ), with this uncertainty bounding set, is quadratically stabilizable.

We now define a set $\mathcal{S}_{nom}$ which characterizes the set of nominally determined quadratic Lyapunov functions. For reasons which will become apparent in the next section, it is convenient to deal with the inverse of the Lyapunov matrix rather than the Lyapunov matrix itself. Hence, we define $\mathcal{S}_{nom}$ as follows: A positive-definite symmetric matrix S is in the set $\mathcal{S}_{nom}$ if there exists an m×n gain matrix K such that $\bar{A}_0 \triangleq A_0 + B_0 K$ has strict left half plane eigenvalues and $\bar{A}_0' S^{-1} + S^{-1}\bar{A}_0$ is negative-definite. (Note that if NOM(Σ) is stabilizable, then the set $\mathcal{S}_{nom}$ will be non-empty.)

The uncertain system (Σ) is said to be *structurally stabilizable via a nominally determined quadratic Lyapunov function* if for each compact $\mathcal{R} \subset R^p$ and each $S \varepsilon \mathcal{S}_{nom}$, the system (Σ) with uncertainty bounding set $\mathcal{R}$, is quadratically stabilizable with Lyapunov function $V(x) = x'S^{-1}x$.

B. Systems With Nonlinear Uncertainty Structure

We now concern ourselves with systems which do not necessarily have linear uncertainty structure. The motivation for our more general definition of structural stabilizability stems from the fact that it is more convenient to work with perturbations in the matrix $\Delta A(r)$ rather than perturbations in the parameter vector r, especially when $\Delta A(r)$ depends nonlinearly on r. Given these potential nonlinear dependencies on the uncertain parameters, there is a possibility that the matrix function $\Delta A(\cdot)$ may be such that $\Delta A(r)$ remains bounded even if $\|r\| \to \infty$. Therefore, an arbitrarily large bound on the uncertain parameter vector r, may not result in arbitrarily large perturbations $\Delta A(r)$. To circumvent this technical difficulty, we associate the system (Σ) with a class of systems with linear uncertainty structure. This class will be denoted SPAN(Σ) and its definition is motivated by one simple fact: Instead of dealing with the admissible set of uncertain parameters $\mathcal{R}$, it is equivalent to deal with the set

$$\Delta A(\mathcal{R}) \triangleq \{\Delta A(r) : r \varepsilon \mathcal{R}\} \subset R^{n \times n}.$$

We now provide a definition.

Given any set of basis matrices $\{A_1, A_2, \ldots, A_n\}$ for the linear space span $\Delta A(R^p)$† $\subset R^{n \times n}$, we can generate a system (Σ_ℓ) in the class SPAN(Σ) as follows: Define a linear matrix function $\Delta A_\ell(\cdot): R^k \to R^{n\times n}$ by

---

†Given any set $G \subset R^n$, span G is defined by span $G \triangleq \{x \varepsilon R^n : x = \sum_{i=1}^{k} \alpha_i g_i$, where $\{g_i\}_{i=1}^{k} \subset G$ and $\alpha_i \varepsilon R$ for $i = 1, 2, \ldots, k\}$.

$$\Delta A_\ell(\alpha) \overset{\Delta}{=} \sum_{i=1}^{k} A_i\alpha_i$$

where α_i is the i^{th} component of the vector $\alpha \varepsilon R^p$. Then the system (Σ_ℓ) is described by the state equation

$$\dot{x}(t) = [A_0 + \Delta A_\ell(\alpha(t))]x(t) + B_0u(t). \qquad (\Sigma_\ell)$$

Given any compact uncertainty bounding set $A \subset R^k$, the vector function $\alpha(\cdot)$ is restricted to be a measurable function such that $\alpha(t)\varepsilon A$ for all $t \geq 0$. We now make use of the class SPAN(Σ) in the following definition.

The system (Σ) is said to be structurally stabilizable if each system (with linear uncertainty structure) $(\Sigma_\ell)\varepsilon$SPAN(Σ) is structurally stabilizable according to the definition given in sub-section IIA.

It follows from the above definition of SPAN(Σ) that the set $\Delta A_\ell(R^k)$ is independent of the system $(\Sigma_\ell)\varepsilon$SPAN(Σ). Hence, it is straightforward to verify that if there exists one system $(\Sigma_\ell)\varepsilon$SPAN(Σ) which is structurally stabilizable, then every system $(\Sigma_\ell)\varepsilon$SPAN(Σ) will be structurally stabilizable.

The system (Σ) is said to be structurally stabilizable via a nominally determined quadratic Lyapunov function if each system (with linear uncertainty structure) $(\Sigma_\ell)\varepsilon$SPAN(Σ) is structurally stabilizable via a nominally determined quadratic Lyapunov function. Again it is straightforward to verify that if there exists one system $(\Sigma_\ell)\varepsilon$SPAN(Σ) which is structurally stabilizable via a nominally determined quadratic Lyapunov function, then every system $(\Sigma_\ell)\varepsilon$SPAN(Σ) will have this property.

C. Notation To Be Used In The Sequel

We let Θ be any matrix whose columns form a set of basis vectors for the linear space

$$N[B_0'] \overset{\Delta}{=} \{x\varepsilon R^n: B_0'x = 0\}.$$

Let V_k denote the inner-product space of symmetric k×k matrices; e.g., see [6]. Given any matrix $M\varepsilon V_k$, $\lambda_{max}[M]$ will denote the maximum eigenvalue of M.

III. THE MAIN RESULT

In this section we present the main result of this paper.

Theorem 3.1: The uncertain system (Σ) is structurally stabilizable via a nominally determined quadratic Lyapunov function if and only if the following conditions hold: (Σ) satisfies matching condition (1.1) and NOM(Σ) is stabilizable.

Before proving this theorem, we first establish some preliminary results.

Our basic concern is to establish the necessity of the matching condition. As far as sufficiency is concerned, we shall use the following Lemma which is an immediate consequence of Theorem 4.1 of [3].

Lemma 3.1: Suppose that the uncertain system (Σ) satisfies matching condition (1.1) and NOM(Σ) is stabilizable. Then the system (Σ) is structurally stabilizable via a nominally determined quadratic Lyapunov function.

Lemma 3.2: The system (Σ) satisfies the matching condition (1.1) if and only if every system (with linear uncertainty structure) $(\Sigma_\ell)\varepsilon$SPAN(Σ) also satisfies this same matching condition.

Proof (Necessity): If the system (Σ) satisfies the matching condition then there exists a continuous matrix function $D(\cdot)$ such that

$$\Delta A(r) = B_0 D(r)$$

for all $r \varepsilon R^p$. Therefore

$$\operatorname{span} \Delta A(R^p) = \operatorname{span} B_0 D(R^p) = B_0 \operatorname{span} D(R^p).$$

Hence, if $\{A_1, A_2, \ldots, A_k\}$ is any set of basis matrices for the space $\operatorname{span} \Delta A(R^p)$, we must have

$$A_i \varepsilon B_0 \operatorname{span} D(R^p)$$

for $i = 1, 2, \ldots, k$. Therefore, there exist matrices $D_1, D_2, \ldots, D_k$ in $\operatorname{span} D(R^p)$ such that

$$A_i = B_0 D_i$$

for $i = 1, 2, \ldots, k$. This implies that, for the system $(\Sigma_\ell) \varepsilon \mathrm{SPAN}(\Sigma)$ corresponding to this set of basis matrices, we have

$$\Delta A_\ell(\alpha) = \sum_{i=1}^{k} B_0 D_i \alpha_i = B_0 \sum_{i=1}^{k} D_i \alpha_i$$

for all vectors $\alpha \varepsilon R^k$. Therefore, if we define the continuous matrix function

$$D_\ell(\alpha) \triangleq \sum_{i=1}^{k} D_i \alpha_i,$$

it is apparent that

$$\Delta A_\ell(\alpha) = B_0 D_\ell(\alpha)$$

for all vectors $\alpha \varepsilon R^k$. Hence any system $(\Sigma_\ell) \varepsilon \mathrm{SPAN}(\Sigma)$ will satisfy the matching condition (1.1).

(Sufficiency): We assume that each system in the class $\mathrm{SPAN}(\Sigma)$ satisfies the matching condition. Now let $(\Sigma_\ell) \varepsilon \mathrm{SPAN}(\Sigma)$ be one such system and note that the matching condition implies that there exists a continuous matrix function $D_\ell(\cdot)$ such that

$$\Delta A_\ell(\alpha) = B_0 D_\ell(\alpha)$$

for all vectors $\alpha \varepsilon R^k$. However, given any matrix $\Delta A \varepsilon \operatorname{span} \Delta A(R^p)$ there exists a vector $\alpha \varepsilon R^k$ such that

$$\Delta A = \Delta A_\ell(\alpha).$$

Therefore, any matrix $\Delta A \varepsilon \operatorname{span} \Delta A(R^p)$ can be written in the form

$$\Delta A = B_0 D$$

where D is a constant $m \times n$ matrix. That is, for each $r \varepsilon R^p$, there exists an $m \times n$ matrix $D(r)$ such that

$$\Delta A(r) = B_0 D(r).$$

Furthermore, using the fact that rank $B_0 = m$, it is clear that the matrix function $D(\cdot)$ is defined uniquely by the expression

$$D(r) = (B_0' B_0)^{-1} B_0' \Delta A(r).$$

The required continuity of D(•) is a consequence of the continuity of $\Delta A(\cdot)$ and moreover,

$$\Delta A(r) = B_0 D(r)$$

for all $r \in R^p$. Hence the matching condition is satisfied. ▯

We shall exploit the following lemma which is established in [2].

Lemma 3.3: Suppose that the uncertain system (Σ) *is quadratically stabilizable with Lyapunov function* $V(x) = x'S^{-1}x$. *Then*

$$\eta'\Theta'(A_0S + SA_0')\Theta\eta + \eta'\Theta'(\Delta A(r)S + S\Delta A'(r))\Theta\eta < 0 \tag{3.1}$$

for all non-zero vectors $\eta \in R^{n-m}$ *and all* $r \in \mathcal{R}$.

Lemma 3.4: The set $\mathcal{S}_{nom}$ *is an open set in the space* $\mathcal{V}_n$.

Proof: Suppose $S_0 \in \mathcal{S}_{nom}$. Then according to the definition of $\mathcal{S}_{nom}$, there exists an m×n matrix K such that $\bar{A}_0 = A_0 + B_0K$ has strict left half plane eigenvalues and the matrix $\bar{A}_0'S_0^{-1} + S_0^{-1}\bar{A}_0$ is negative-definite. Therefore, the matrix $\bar{A}_0S_0 + S_0\bar{A}_0'$ is also negative-definite and it follows that

$$\lambda_{max}[\bar{A}_0S_0 + S_0\bar{A}_0'] < 0.$$

Now, using the continuity of the $\lambda_{max}[\cdot]$ function, it follows that there exists a constant $\delta_1 > 0$ such that if $S \in \mathcal{V}_n$ satisfies $\|S - S_0\| < \delta_1$, then $\bar{A}_0S + S\bar{A}_0'$ is negative-definite. Furthermore, we note that the matrix S_0 is positive-definite and the set of positive-definite matrices is an open set in the space $\mathcal{V}_n$; e.g. see [6]. Therefore, there exists a second constant $\delta_2 > 0$ such that if $S \in \mathcal{V}_n$ satisfies $\|S - S_0\| < \delta_2$, then S is a positive-definite matrix. Hence, $\|S - S_0\| < \min\{\delta_1, \delta_2\}$ implies that $S \in \mathcal{S}_{nom}$. Therefore, $\mathcal{S}_{nom}$ is an open set in the space $\mathcal{V}_n$. ▯

Lemma 3.5: Suppose that the system (Σ) *is structurally stabilizable via a nominally determined quadratic Lyapunov function. Then*

$$\text{span}\ \mathcal{S}_{nom} = \mathcal{V}_n.$$

Proof: We recall that $\Delta A(r) = 0$ for some $r \in \mathcal{R}$. This, together with the fact that the uncertain system (Σ) is structurally stabilizable implies that the system NOM(Σ) is stabilizable. Therefore, the set $\mathcal{S}_{nom}$ is non-empty. Lemma 3.4 states that the set $\mathcal{S}_{nom}$ is an open set. Now, given any matrix $W \in \mathcal{V}_n$ and a matrix $X \in \mathcal{S}_{nom}$, one can choose $\delta > 0$ sufficiently small so as to ensure that

$$Z \triangleq X + \delta W \in \mathcal{S}_{nom}.$$

Therefore,

$$W = \frac{Z}{\delta} - \frac{X}{\delta} \in \text{span}\ \mathcal{S}_{nom}.$$

It now follows that

$$\text{span}\ \mathcal{S}_{nom} = \mathcal{V}_n. \ ▯$$

Lemma 3.6: Suppose that the system (Σ) *is structurally stabilizable via a nominally determined quadratic Lyapunov function. Then, given any system* $(\Sigma_\ell) \in \text{SPAN}(\Sigma)$, *the corresponding set of basis matrices* $\{A_1, A_2, \ldots, A_k\}$ *satisfies the following condition:*

$$\Theta'[A_iS + SA_i']\Theta = 0 \tag{3.2}$$

for all $S\varepsilon\mathcal{V}_n$ *and* $i=1,2,\dots,k$.

Proof: We will first establish equation (3.2) for all matrices $S\varepsilon\mathcal{S}_{nom}$. Subsequently, it will be shown that equation (3.2) must hold for all matrices $S\varepsilon\mathcal{V}_n$.

Since the system (Σ) is structurally stabilizable via a nominally determined quadratic Lyapunov function, it is apparent that all systems in the class SPAN(Σ) must also have this property. Indeed, let (Σ_ℓ) be any system in the class SPAN(Σ). Then, for each compact set $\mathcal{A}\subset R^k$ and each matrix $S\varepsilon\mathcal{S}_{nom}$, the system (Σ_ℓ) with uncertainty bounding set $\mathcal{A}$ is quadratically stabilizable with Lyapunov function $V(x)=x'S^{-1}x$. We now apply Lemma 3.3 and infer the following: Given any compact set $\mathcal{A}\subset R^k$ and matrix $S\varepsilon\mathcal{S}_{nom}$

$$\eta'\Theta'[A_0S+SA_0']\Theta\eta+\eta\Theta'[\Delta A_\ell(\alpha)S+S\Delta A_\ell'(\alpha)]\Theta\eta< 0$$

for all $\alpha\varepsilon\mathcal{A}$ and all non-zero vectors $\eta\varepsilon R^{n-m}$. Equivalently

$$\eta'\Theta'[A_0S+SA_0']\Theta\eta+\max_{\alpha\varepsilon\mathcal{A}}\{\eta'\Theta'[\Delta A_\ell(\alpha)S+S\Delta A_\ell'(\alpha)]\Theta\eta\}<0 \tag{3.3}$$

for all non-zero vectors $\eta\varepsilon R^{n-m}$. In particular, if we take the set $\mathcal{A}$ to be the hypercube $\{\alpha\varepsilon R^k\colon\ \alpha=(\alpha_1,\alpha_2,\dots,\alpha_k)',\ \max_i|\alpha_i|\le\bar{\alpha}\}$ and use the linear structure of $\Delta A_\ell(\cdot)$, inequality (3.3) leads to the following statement: Given any matrix $S\varepsilon\mathcal{S}_{nom}$, any non-zero vector $\eta\varepsilon R^{n-m}$ and any $\bar{\alpha}>0$,

$$\eta'\Theta'[A_0S+SA_0']\Theta+\bar{\alpha}\sum_{i=1}^{k}|\eta'\Theta'[A_iS+SA_i']\Theta\eta|<0 \tag{3.4}$$

where $\{A_1,A_2,\dots,A_k\}$ is the set of basis matrices corresponding to the system (Σ_ℓ). Since $\bar{\alpha}$ can be arbitrarily large, the only way inequality (3.4) can hold is if

$$\Theta'[A_iS+SA_i']\Theta = 0 \tag{3.5}$$

for all $S\varepsilon\mathcal{S}_{nom}$ and $i=1,2,\dots,k$.

The next part of this proof involves extending equation (3.5) to all matrices $S\varepsilon\mathcal{V}_n$. For each integer $i=1,2,\dots,k$, we define a linear operator $\mathcal{A}_i\colon\mathcal{V}_n\to\mathcal{V}_{n-m}$ by

$$\mathcal{A}_i(S)\triangleq\Theta'[A_iS+SA_i']\Theta.$$

Let $\mathcal{N}[\mathcal{A}_i]$ denote the nullspace of the operator $\mathcal{A}_i$. Using this notation and (3.5), it is clear that

$$\mathcal{S}_{nom}\subseteq\mathcal{N}[\mathcal{A}_i]$$

for $i=1,2,\dots,k$. Hence,

$$\mathcal{S}_{nom}\subseteq\bigcap_{i=1}^{k}\mathcal{N}[\mathcal{A}_i]$$

Note, however, that $\bigcap_{i=1}^{k}\mathcal{N}[\mathcal{A}_i]$ is a subspace of $\mathcal{V}_n$ and hence

$$\text{span}\ \mathcal{S}_{nom}\subseteq\bigcap_{i=1}^{k}\mathcal{N}[\mathcal{A}_i]. \tag{3.6}$$

Invoking Lemma 3.5 we can replace (3.6) by

$$\mathcal{V}_n\subseteq\bigcap_{i=1}^{k}\mathcal{N}[\mathcal{A}_i].$$

Therefore,

$$\Theta'[A_i S + S A_i']\Theta = 0$$

for all $S \varepsilon \mathcal{V}_n$ and $i = 1,2,\ldots,k$. □

Lemma 3.7: Suppose that the system (Σ) is structurally stabilizable via a nominally determined quadratic Lyapunov function. Then given any system (Σ_ℓ)εSPAN(Σ), the corresponding set of basis matrices $\{A_1, A_2, \ldots, A_k\}$ satisfies the following condition:

$$\Theta' A_i = 0 \tag{3.7}$$

for $i = 1,2,\ldots,k$.

Proof: We proceed by contradiction. Suppose that equation (3.7) does not hold for some $i \varepsilon \{1,2,\ldots,k\}$. Then, if we let θ_j denote the jth column of matrix Θ, there exists some $j \varepsilon \{1,2,\ldots,n-m\}$ such that

$$c_j \overset{\Delta}{=} \theta_j' A_i \neq 0.$$

Let

$$N[c_j'] \overset{\Delta}{=} \{x \varepsilon R^n : c_j' x = 0\}$$

and

$$N[\theta_j'] \overset{\Delta}{=} \{x \varepsilon R^n : \theta_j' x = 0\}.$$

Since $c_j \neq 0$ and $\theta_j \neq 0$, the two sets $N[c_j']$ and $N[\theta_j']$ are (n-1)-dimensional subspaces of R^n. Therefore, the set $N[c_j'] \cup N[\theta_j'] \neq R^n$. Hence, there there exists a vector $v \varepsilon R^n$ such that $v' c_j \neq 0$ and $\theta_j' v \neq 0$. We now define the symmetric matrix

$$S^* \overset{\Delta}{=} \frac{vv'}{v' c_j} .$$

Notice that this matrix has the property that

$$v = S^* c_j$$

and moreover, the (j,j)-th element of the matrix $\Theta'[A_i S^* + S^* A_i']\Theta$ is

$$\theta_j'[A_i S^* + S^* A_i']\theta_j = 2\theta_j' S^* c_j = 2\theta_j' v \neq 0.$$

This implies that the matrix $\Theta'[A_i S^* + S^* A_i']\Theta$ is non-zero which contradicts Lemma 3.6. □

We are now in a position to prove the main result of this paper.

Proof of Theorem 3.1 (Sufficiency): To establish sufficiency, we assume that the system (Σ) satisfies the matching condition (1.1) and that NOM(Σ) is stabilizable; it must be shown that the system (Σ) is structurally stabilizable via a nominally determined quadratic Lyapunov function. Indeed, let (Σ_ℓ) be any system in the class SPAN(Σ). Since the system (Σ) satisfies the matching condition, Lemma 3.2 states that (Σ_ℓ) will also satisfy the matching condition. Since NOM(Σ_ℓ) = NOM(Σ), stabilizability of NOM(Σ) implies stabilizability of NOM(Σ_ℓ). Invoking Lemma 3.1, it follows that the system (Σ_ℓ) is structurally stabilizable via a nominally determined quadratic Lyapunov function. However, recalling that (Σ_ℓ) is an arbitrary system in the class SPAN(Σ), it must follow that the system (Σ) is structurally stabilizable via a nominally determined quadratic Lyapunov function.

(Necessity): To establish necessity, we suppose that the system (Σ) is structurally stabilizable via a nominally determined quadratic Lyapunov function; it must be shown that the system (Σ) satisfies the matching condition (1.1) and that NOM(Σ) is stabilizable. Taking note of Lemma 3.5, it is clear that structural stabilizability of (Σ) implies that the system NOM(Σ) is stabilizable.

Next, let $(\Sigma_\ell) \varepsilon \mathrm{SPAN}(\Sigma)$ be a given system with linear uncertainty structure. It follows from Lemma 3.7 that

$$\Theta' A_i = 0 \tag{3.8}$$

for $i = 1,2,\ldots,k$. Recalling that rank $B_0 = m$ and that the columns of Θ span $N[B_0']$, it is clear that the $n \times n$ matrix $[B_0 \vdots \Theta]$ is non-singular. Let the inverse of this matrix be partitioned as

$$\begin{bmatrix} F \\ \hline G \end{bmatrix} = \begin{bmatrix} B_0 & \vdots & \Theta \end{bmatrix}^{-1}$$

where F is an $m \times n$ matrix and G is an $(n-m) \times n$ matrix. As a consequence of this partitioning, it is apparent that

$$B_0 F + \Theta G = I \tag{3.9}$$

and

$$G B_0 = 0. \tag{3.10}$$

Letting g_j' denote the j^{th} row of G, (3.10) is tantamount to

$$B_0' g_j = 0$$

for $j = 1,2,\ldots,n-m$. Therefore, $g_j \varepsilon N[B_0']$ for $j = 1,2,\ldots,n-m$ and hence, each g_j can be written in the form $g_j = \Theta h_j$ for appropriate $h_j \varepsilon R^{n-m}$. This implies that the matrix G can be expressed as

$$G = H\Theta' \tag{3.11}$$

where H has the vectors h_j, $j = 1,2,\ldots,n-m$ as its rows.

For each $i \varepsilon \{1,2,\ldots,k\}$, we define the matrix

$$D_i \stackrel{\Delta}{=} FA_i.$$

We now claim that $A_i = B_0 D_i$ for $i = 1,2,\ldots,k$. Indeed, let $i \varepsilon \{1,2,\ldots,k\}$ be fixed. It now follows from equation (3.9) that

$$B_0 F A_i + \Theta G A_i = A_i$$

and combining this with equations (3.8) and (3.11), we obtain

$$A_i = B_0 F A_i = B_0 D_i.$$

Therefore, for the system (Σ_ℓ), the function $\Delta A_\ell(\cdot)$ is given by

$$\Delta A_\ell(\alpha) = \sum_{i=1}^{k} A_i \alpha_i = B_0 \sum_{i=1}^{k} D_i \alpha_i \stackrel{\Delta}{=} B_0 D_\ell(\alpha)$$

where α_i denotes the i^{th} component of the vector $\alpha \varepsilon R^k$.

We can now conclude that the system (Σ_ℓ) satisfies the matching condition. By applying Lemma 3.2 it follows that the system (Σ) also satisfies the matching condition. □

ACKNOWLEDGMENT

Discussions with Professor B.R. Barmish and Mr. C.V. Hollot are gratefully acknowledged.

REFERENCES

[1] G. Leitmann, "Guaranteed Asymptotic Stability for some Linear Systems with Bounded Uncertainties," ASME Journal of Dynamical Systems, Measurement and Control, Vol. 101, No. 3 1979.

[2] C.V. Hollot and B.R. Barmish, "Optimal Quadratic Stabilizability of Uncertain Linear Systems," Proceedings of the 18th Allerton Conference on Communications, Control and Computing, University of Illinois, Monticello, 1980.

[3] J.S. Thorp and B.R. Barmish, "On Guaranteed Stability of Uncertain Linear Systems via Linear Control," Journal of Optimization Theory and Applications, Vol. 35, No. 4, 1981.

[4] B.R. Barmish, "Necessary and Sufficient Conditions for Quadratic Stabilizability of an Uncertain Linear System," in preparation.

[5] B.R. Barmish, "Stabilization of Uncertain Systems via Linear Control," Proceedings of the IEEE Asilomar Conference on Computers, Circuits and Systems, Monterey, 1981.

[6] A. Berman, Cones, Matrices and Mathematical Programming, Springer-Verlag, Berlin, 1973.

A FILTERING APPROACH TO TRANSIENT FAULT DIAGNOSIS

M. A. KENNEDY and G.G.L.MEYER
Electrical Engineering and Computer Science Department
The Johns Hopkins University
Baltimore, MD 21218

SUMMARY

This paper describes a new approach to transient fault diagnosis for complex devices. The ephemeral nature of transient faults and the difficulty of performing a large number of tests in a short time interval result in incomplete information for diagnosis. In this approach a diagnosis is obtained by applying recursive filtering techniques to these incomplete test results. Therefore, this recursive filtering approach to fault diagnosis is analogous to the techniques used in signal processing to extract information from noisy or incomplete data. The approach addresses both permanent and transient faults and provides the capability to diagnose using incomplete test results while reducing the average time required for diagnosis.

The areas of emphasis of existing approaches that address the transient fault problem may be divided into several categories: probabilistic models [BRE73, SAV78, VAR81], design of testing procedures [LIA81, SAV80, SPI79], conditions for diagnosability [MAL78, MAL80], and simulation of transient fault effects [BAL69]. In the first two categories, the fault diagnosis problem is investigated at the circuit or gate-level. The third category contains system-level models, while the fourth is concerned with the statistics of a particular system.

The drawbacks of these existing approaches include: the difficulty in acquiring probability measures for large numbers of faults, gate-level analysis is not suited to complex devices, repetitive or continuous testing that implies that testing must be a completely separate function from normal operation, and waiting for an entire set of tests to be completed or disregarding test results that can not be attributed to a permanent fault situation.

To remedy the difficulties inherent in the existing works, the new approach will possess the following features: it will be a system-level approach using both concurrent and functional testing, have a weak dependence on probability, work for solid as well as transient faults, and use test information as it is obtained. The need to consider VLSI devices

at the outset that the approach must be at the system level. Functional testing implies that the primary objective of testing is to evaluate if a device is performing its function correctly, rather than finding the precise location and type of fault. Concurrent testing implies that the tests are run concurrently with normal system operation, and that not all the test may be run at the same time: this produces the partial syndromes or test results which are the basis of the filtering approach. Concurrent testing has some of the advantages of continuous testing without requiring the system to be completely dedicated to testing. The weak dependence on probability minimizes the need for quantities which are difficult to obtain while allowing the use of relative probability measures when available. The use of test information as it is obtained is inherent in the concept of the partial syndrome.

The proposed approach uses four basic elements: a system model, a fault model, the concept of partial syndromes, and the concept of filtered syndromes. The system model is concerned with the basic constitutive units of the system, and the testing interconnection structure that represents the tests performed on the units in the system. The fault model relates the test outcomes to the faults that occur in the system. Partial syndromes are defined as the collection of test results s(t), observed at time t, t = 1,2,... , and consist of elements in the set $Z = \{0,1,e\}$, where a 0 indicates the test passed, a 1 indicates the test failed, and an e indicates that no information is available about that test. Filtered syndromes, $s_f(t)$, are related to the partial syndromes and time t by a may, $g(.,.)$, that is, $s_f(t) = g(s,\tau)$, $\tau = 1,2,...5$. The filtering process defines how the partial syndromes are used to update the filtered syndrome.

REFERENCES

1. BAL69 Ball, M., and Hardie, F., Effects and Detection of Intermittent Failures in Digital Systems, AFIPS Conference Proceedings, Vol. 35, 1969, pp. 329-335.

2. BRE73 Breuer, M.A., Testing for Intermittent Faults in Digital Circuits, IEEE Transactions on Computers, Vol. C-22, No. 3 March 1973, pp. 241-246.

3. Kar79 Karunanithi, S. and Friedman, A.D., Analysis of Digital Systems Using a New Measure of System Diagnosis, IEEE Transactions on Computers, Vol. C-28, No. 2, February 1979, pp. 121-133.

4. LIA81 Liaw, C.C., Su, S.Y.II., and Malaiya, Y.K., Test-Experiments for Detection and Location of Intermittent Faults in Sequential Circuits, IEEE Transactions on Computers, Vol. C-30, No. 12, December 1981, pp. 989-995.
5. MAL78 Mallela, S., and Masson, G.M., Diagnosable Systems for Intermittent Faults, IEEE Transactions on Computers, Vol. C-27, No. 6, June 1978, pp. 360-366.
6. MAL80 Mallela, S., and Masson, G.M., Diagnosis Without Repair for Hybrid Fault Situations, IEEE Transactions on Computers, Vol. C-29, No. 6, June 1980, pp. 461-470.
7. PRE67 Preparata, F.P., Metze, G., and Chien, R.T., On the Connection Assignment Problem of Diagnosable Systems, IEEE Transactions on Electronic Computers, Vol. EC-16, December 1967, pp. 848-854.
8. SAV78 Savir, J., Model and Random-Testing Properties of Intermittent Faults in Combinational Circuits, Journal of Design Automation and Fault Tolerant Computing, No. 3, July 1978, pp. 215-230.
9. SAV80a Savir, J., Testing for Single Intermittent Failures in Combinational Circuits by Maximizing the Probability of Fault Detection, IEEE Transactions on Computers. Vol. C-29, No. 5, May 1980, pp. 410-415.
10. SAV80b Savir, J., Detection of Single Intermittent Faults in Sequential Circuits, IEEE Transactions on Computers, Vol. C-29, No. 7, July 1980, pp. 673-678.
11. SPI79 Spillman, R.J., An Analysis of an Intermittent Fault Detection Procedure, Journal of Design Automation and Fault Tolerant Computing, Vol. 3, No. 3-4, Winter 1979, pp. 227-239.
12. VAR81 Varshney, P.K., On Analytical Modeling of Intermintent Faults in Digital Systems, IEEE Transactions on Computers, Vol. C-28, No. 10, October 1979, pp. 786-791.

This work was supported by the Office of Naval Research under Contract N00014-80-C-0772.

FAULT LOCATION IN SEQUENTIAL TREES

S. CHANDRA
Department of Mines and Energy
Darwin, N.T. 5790

KEWAL K. SALUJA
Department of Electrical and Computer Engineering
University of Newcastle, N.S.W. 2308 Australia.

ABSTRACT

Tree structures occur very frequently in software as well as in hardware systems. In this paper we study the diagnostic properties of certain hardware tree structures. First we study a tree consisting of AND-Exclusive OR-Delay (AND-EOR-DELAY) modules, which can be used to realize an arbitrary sequential machine. We assume the tree to be uni-directional and the inputs to the tree are applied from the leaves and the output is observed at the root of the tree. The fault model is assumed to be permanent stuck-type faults only. Some other interesting applications of our results in testing search trees, multiplier trees etc. are also discussed.

1. INTRODUCTION

Tree structures occur very frequently in software as well as in hardware systems [1-7]. Modular tree realizations of sequential machines using AND-OR-DELAY modules (Fig. 1a) have been reported in literature [1,4,5]. However, either AND-OR-DELAY or AND-EOR-DELAY (Fig. 1b) modules can be used to realize any single input n-definite machine [3,4] by a tree of depth n as shown in Fig. 2(a). Any arbitrary sequential machine can be realized by a universal tree structure of Fig. 2(b) [3,4,5].

Tree structures formed by AND-OR-DELAY modules as basic modules, have been studied for diagnosis of single and multiple faults [2,9,10]. The diagnostic tests for such circuits are exhaustive or semi-exhaustive in nature and often assume the accessibility of all the control inputs. More recently, Saluja [3,8] has shown that modular trees of AND-EOR-DELAY modules are relatively easier to test as far as single faults are concerned. In this paper, the problem of locating single stuck-at faults in such trees is investigated. Some preliminaries, including the fault model, are presented in Section 2. In Section 3 single fault location tests for AND-EOR combinational trees are derived. These results are then used in Section 4 to derive sequences to test sequential machines. In Section 5 some interesting applications of our results are pointed out.

2. PRELIMINARIES

For the derivation of fault location tests the approach adopted here is similar to the one adopted by Saluja [3]. Therefore a familiarity with that work [3] will be assumed. The basic idea being that in order to derive diagnostic tests for a sequential tree one needs to study an equivalent combinational tree of Fig. 3. The tree of Fig. 3 is made up of AND-EOR modules; modules of Fig. 1(b) without the delays. In this paper single input/single output synchronous sequential machines are considered. The results can be readily extended to multiple input/multiple output machines.

The fault model considered here assumes (i) the presence of stuck-at faults (s-a-1 / s-a-o) only, (ii) input/output faults for Exclusive-OR gates and (iii) that the primary inputs (bus bars) to the combinational tree of Fig. 3 to be fault free. Although (ii) is a restrictive assumption it is believed that only a slight modification of the results would be required to include faults within Exclusive-OR gates. The faults in the primary inputs can be diagnosed easily by little additional logic and/or additional observation points.

In the tree of Fig. 3 the output of a module j in level i can be written as

$$c^{i-1}_{j-1} = g(x_n, \ldots, x_{i+1}) \oplus x_i h(x_n, \ldots, x_{i+1})$$

or more precisely as

$$c^{i-1}_{j-1} = c^{i}_{2j-2} \oplus x_i c^{i}_{2j-i} \tag{1}$$

If (1) is used recursively then c^{i-1}_{j-1} can be expressed as

$$c^{i-1}_{j-1} = c^{n}_{k} \oplus c^{n}_{k+1} x_n \oplus \ldots \oplus c^{n}_{k+2^{n-i+1}-1} x_n \ldots x_{i+1} \tag{2}$$

where $k = 2^{n-i+1}(j-1)$.

It is to be noted that the control inputs which appear in (2) are the control inputs of a subtree, shown in Fig. 3, with output c^{i-1}_{j-1}. Clearly the output of each module in the tree is the output of a unique subtree of the type described above. The subtree associated with a module m would be referred to as the 'shadow' of m and written as $S(m)$. Fig. 3 shows a module j and its associated shadow $S(j)$.

The following four vectors, defined below, are of special interest to us.

$$\underline{C}^{i} = (c^{i}_{0}, c^{i}_{1}, \ldots, c^{i}_{2^{i}-1}) = (1, 1, \ldots, 1) \triangleq \underline{1}$$

$$\underline{C}^{i} = (c^{i}_{0}, c^{i}_{1}, \ldots, c^{i}_{2^{i}-1}) = (0, 0, \ldots, 0) \triangleq \underline{0}$$

If $x_n = x_{n-1} = \ldots, x = 1$ then the tree of Fig. 3 reduces to an EOR tree. Two test vectors $\underline{C}^{n} = \underline{P}$ and $\underline{C}^{n} = \underline{Q}$ can then be derived [6] such that $\underline{P}$ and $\underline{Q}$ together assure an input of $c^{i}_{2j} = 0$ and $c^{i}_{2j+1} = 1$ for a module j in level i.

The following lemma holds for a fault free tree.

Lemma 1 [8]: (a) If $\underline{C}^{i} = \underline{1}$ and $x_i = 1$ then $\underline{C}^{i-1} = \underline{0}$

(b) If $\underline{C}^{i} = \underline{1}$ and $x_i = 0$ then $\underline{C}^{i-1} = \underline{1}$

(c) If $\underline{C}^{i} = 0$ then $\underline{C}^{i-1} = \underline{0}$

Note that only $\underline{C}^{n}$ control inputs are accessible for testing purposes.

3. FAULT LOCATION IN COMBINATIONAL TREES

In this section a constructive procedure for the investigation of a fault location test set is given. Faults are divided into different classes and tests for the location of each class of faults are derived. From the union of all such tests a complete single fault location test set is then obtained. The procedure presented here is based on the fact that for any two distinct single faults, with output functions f_1 and f_2, if a test fector t can be found such that

$$(f_1 \oplus f_2)(t) = 1$$

then the two faults can be distinguished from each other. Clearly, if each single fault can be distinguished from every other fault in the network, by some test in a test set T then T must be a single fault location test set. However a fault can only be located up to its equivalence class. For example an AND gate input s-a-o cannot be distinguished from its output s-a-o etc. If a fault is in module m of a tree then f_i represents the faulty output.

Definition 1: Two modules m_1 and m_2 are said to be 'compatible' if $m_2 \notin S(m_1)$ and $m_1 \notin S(m_2)$. In Fig. 4 modules m_1 and m_2 are compatible whereas m_2 and m_i are incompatible.

Definition 2: An input x_i is a blocking input for a module m if $x_i = 0$ desensitizes the path from m to the output. Clearly a given module may or may not have any blocking input(s).

Lemma 2: For any compatible modules m_1 and m_2 there exists an input x_i such that x_i is a blocking input for either m_1 or m_2 but not for both m_1 and m_2.

Proof: Since m_1 and m_2 are compatible the path from $m_1(m_2)$ to the output does not pass through $m_2(m_1)$. The paths, from m_1 to the output, and m_2 to the output, however, meet at some module m_3 of the tree. It is evident that one of the paths will join m_3 at its AND gate input and the other at the EOR input. If the primary input to m_3 is x_i then $x_i = 0$ will block only one of the paths. ∇∇∇

A. DISTINGUISHABILITY OF FAULTS WITHIN A MODULE:

Lemma 3: A single fault in an AND-EOR module can be located by the following test set T_0:

$$T_0 = \left[\begin{array}{ccc} x & c_1 & c_0 \\ 0 & 0 & 0 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \\ 1 & 0 & 0 \end{array}\right]$$

Proof: The above test set T_0 can be verified to be a fault location test set by constructing a fault table [11]. ∇∇∇

Corollary 1: Two faults within a module m of a tree can be distinguished.

Proof: Let m be a module in the ith level of a tree of depth n. To ensure that all the tests in T_0 are applied to m, choose (i) $x_{i-1} = x_{i-2} = \ldots = x_1 = 1$ thus sensitizing the path from m to the output (ii) $\underline{C}^n$ from the set $\{\underline{1}, \underline{0}, \underline{P}, \underline{Q}\}$ and (iii) the appropriate values of the primary inputs from $x_n, x_{n-1}, \ldots, x_{i+1}$ with $x_i = 0, 1$. Since T_0 is a location test set for m two faults within a module in a tree can be distinguished.

∇∇∇

B. DISTINGUISHABILITY OF FAULTS IN COMPATIBLE MODULES:

Consider now the distinguishability of inter-modular faults. The following lemma is stated without proof for space limitation.

Lemma 4: If m_1 and m_2 are two compatible modules then there exists an input t such that

$$(f_1 \oplus f_2)(t) = 1$$

C. DISTINGUISHABILITY OF FAULTS IN INCOMPATIBLE MODULES:

Let m_1 be a module, with fault f_1, which lies within the shadow, $S(m_2)$, of a module m_2 with fault f_2. The distinguishability of f_1 from f_2 is based on finding two inputs t_1 and t_2 such that both t_1 and t_2 desensitize the output of m_1 and generate a 0 and a 1 (or a 1 and a 0) respectively as outputs of m_1, i.e.

$$f_1(t_1) = f'(t_1) = 0 \quad \text{and} \quad f_2(t\ ) = f'(t_2) = 1$$

where f' is the fault free output of the subtree $S(m_1)$. The tests t_1 and t_2 are made up of the control inputs $\underline{C}^n$ and the primary inputs from nth level to the level containing m_1. The problem of distinguishing f_1 from f_2 is now reduced to detecting f_2. This is achieved by sensitizing the path from m_2 to the output.

Lemma 5: If the output module m_0 is faulty but the output of the tree is not s-a-1/0 then there exist inputs t_1 and t_2 such that

$$f_0(t_1) = f(t_1) = 1 \quad \text{and} \quad f_0(t_2) = f(t_2) = 0$$

$f(t_1)$ and $f(t_2)$ are the fault free outputs of the network for inputs t_1 and t_2 respectively.

Proof: Fig. 4 shows the output module m_0 and its two associated subtrees S_1 and S_2. As S_1 and S_2 are fault free any arbitrary output at the outputs of S_1 and S_2 can be generated by choosing the primary inputs and the control inputs appropriately. For example if input α to m_0 (Fig. 4) is s-a-1 then by choosing $x_1 = 1$ and the other inputs such that the outputs of S_2 and S_1 are 11 and 10. In the first case the result is $f_0(t_1) = f(t_1) = 0$ and in the second case $f_0(t_2)=f(t_2) = 1$.

∇∇∇

Corollary 2: If the output of a tree is not s-a-1/0 then for any fault in a module m_1 of a tree there exist inputs t_1 and t_2 such that

$$f_1(t_1) = f(t_1) = 1 \quad \text{and} \quad f_1(t_2) = f(t_2) = 0.$$

Proof: Choose all the primary inputs from m_1 to the output such that the path from m_1 to the output is sensitized and then apply Lemma 5 for all the other inputs.

∇∇∇

In the proof of Lemma 5 no restriction was imposed on the control inputs. However, it is a simple matter to check that Lemma 5 holds if control inputs are restricted to the set $\{\underline{0}, \underline{1}, \underline{P}, \underline{Q}\}$.

D. SINGLE FAULT LOCATING TEST SET:

The following theorem can now be stated.

Theorem 1: AND-EOR trees are single fault locatable.

Proof: This theorem will be proved by showing that for any two faults f_1 and f_2 there exists a test input t such that $(f_1 \oplus f_2)(t) = 1$.

Case 1: f_1 and f_2 are in the same module - this case has been dealt within Corollary 1.

Case 2: f_1 and f_2 are in two distinct compatible modules - this case has been dealt within Lemma 4.

Case 3: f_1 and f_2 are such that the module with fault f_1 is in the shadow of the module with fault f_2. Consider two nonintersecting subtrees U_1 and U_2 associated with the module with fault f_2 (Fig. 4). If the output of U_1 or U_2 is s-a-1/ or 0 then this case reduces to case 1. Otherwise by corollary 2, there exist inputs such that the effect of the fault f_1 is desensitized and any desired input can be applied to the module with fault f_2. The path from f_2 to the output can now be sensitized i.e., $f_1(t) = f(t)$ and $f_2(t) \neq f(t)$ for some input t.

∇∇∇

Definition 3: Let A be a binary n-tuple. A can be written as $A = A_1A_2$ such that A_1 is a k_1-tuple consisting of all zeros and A_2 is a k_2-tuple with a nonzero left most element $(k_1+k_2 = n)$. If x_1 and x_2 are the number of zeros in A_1 and A_2 respectively then A is said to belong to a 2-tuple (x_1, x_2).

Example: For $n = 5$ and $A = 00110$. $A_1 = 00$ and $A_2 = 110$, thus $x_1 = 2$ $x_2 = 1$ and therefore $A \in (2,1)$.

Only other 5-tuple which belongs to (2,1) is 00101.

Define the following test sets for an n-level AND-EOR tree.

$T_1 = \{<\underline{1}\ A>/A \in (0, 0)\}$ $\quad T_2 = \{<\underline{1}\ A>/A \in (0, 1)\}$

$T_3 = \{<\underline{0}\ A>\ A \in (0, 0)\}$ $\quad T_4 = \{<\underline{0}\ A>\ A \in (0, 1)\}$

$T_6 = \{<\underline{0}\ A>\ A \in (1, 0)\}$ $\quad T_7 = \{<\underline{P}\ A>\ A \in (0, 0)\}$

$T_8 = \{<\underline{1}\ A>\ A \in (i, 0)\}$ for $i = 1, 2, 3, \ldots, n$

$T_9 = \{<\underline{1}\ A>\ A \in (i, 1)\}$ for $i = 1, 2, \ldots, n-2$

Theorem 2: The test set $T = T_1 \cup T_2 \cup T_3 \ldots \cup T_9$ is a single fault location test set for an n level AND-EOR combinational tree. The cardinality of T is $n + 5 + \frac{(n+2)(n-1)}{2}$.

Proof: T contains all the necessary blocking inputs and $\underline{C}^n$ inputs discussed earlier. Therefore T is a single fault location test set.

The cardinality of the set T can be worked out from the fact that

$$|T_1| = |T_3| = |T_5| = |T_6| = |T_7| = 1; \quad |T_2| = |T_4| = n-1;$$

$$|T_8| = n, \quad |T_9| = \frac{(n-1)(n-2)}{2}$$

∇∇∇

Theorem 3: The test set T of theorem 2 can detect all faults of multiplicity two.

Proof: This result follows from the proof of Lemmas 4 and 5.

∇∇∇

4. DEFINITE TREES AND SEQUENTIAL MACHINES

Tests for locating a single fault in AND-EOR combinational trees derived in Section 3 can be sequentialized by using the approach taken by Saluja [8, Theorem 6].

A. DEFINITE TREES: The tests given in Theorem 2 can be sequentialized without exploiting any of the overlapping property of n-tuples, thus generating a test sequence of length $\approx n^3$. This sequence can be used to dianose an n-definite tree consisting of AND-EOR-DELAY modules. For the reduction of the control inputs the procedure given by Saluja [3] can be applied without any modification.

B. SEQUENTIAL MACHINES: By introducing some additional control inputs in the feedback path the modular realization of sequential machines (Fig. 2(b)) can be divided into two definite trees. Both the definite trees so obtained can then be diagnosed simultaneously be the same tests. Since each module in these trees has 2-inputs (i.e. 4 control inputs) a small modification in the test sequence will be required [3].

C. MULTIPLE INPUTS/OUTPUTS MODULES: The results of the previous section can be generalized to include multiple input/output modules on the same lines as discussed in [3].

5. ADDITIONAL APPLICATIONS

Some interesting application of the results given in Section 3 is the use of the test sequence to test search trees [6, pp. 442], multiplier trees [7] etc. For example search tree can be seen as a uni-directional tree, but it receives its input from the root as opposed to the leaves. Each module will be a multiple-input/output module with multi-bit storage. The x input lines will become the additional control lines for such a tree and thus the test sequence will be applied from the root and enabling the controls to transfer the input to appropriate leaves. Thus the tree could be tested by observing the leaves.

In case of bidirectional trees, these results could be applied in straightforward manner. However, in such a case validity of the fault model will depend on the realization of the modules.

6. CONCLUSION

In this paper we have studied the problem of single fault location in modular tree realization of sequential machines. The test length given in Section 3 can be reduced to the order of n^2 if the test engineer uses his/her ingenuity to sequentialize n-tuples. A systematic procedure to obtain a minimal length test sequence is currently under investigation.

Although, the fault model chosen by us is very restrictive, we believe that the test set given in Theorem 2 will diagnose any arbitrary single fault in the tree.

We must also point out that only a subset of the test given in Theorem 2 can detect all single and double faults in the tree.

7. ACKNOWLEDGEMENT

Support from RRB Australia is thankfully acknowledged.

REFERENCES

[1] R.A. Thompson and F.G. Gray, "Universal Modular Trees: A Design Procedure", *IEEE Transactions on Computers,* Vol. C-27, pp. 53-63, January 1978.

[2] F.G. Gray et al., "Diagnosis of Faults in Modular Trees", *IEEE Trans. Comput.,* Vol. C-28, pp. 342-353, May 1979.

[3] K.K. Saluja, "Synchronous Sequential Machines: A Modular and Testable Design", *IEEE Trans. Comput.,* Vol. C-29, pp. 1020-1025, Nov. 1980.

[4] T.F. Arnold, C.J. Tan and M.M. Newborn, "Iteratively Realized Sequential Circuits", *IEEE Trans. Comput.,* Vol. C-19, pp. 54-66, Jan. 1970.

[5] M.M. Newborn, "A Synthesis Technique for Binary Input-Output Synchronous Sequential Moore Machine", *IEEE Trans. Comput.,* Vol. C-17, July 1968.

[6] Knuth, D.E., The Art of Computer Programming: Sorting and Searching, Vol. 3, Addison Wesley, 1973.

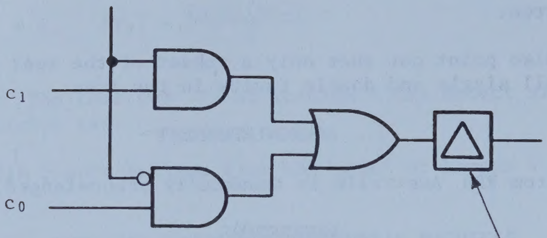
[7] Fuchs, H. and Poulton, J., "Pixel Planes: A VLSI-Oriental Design for a Raster Graphic Engine", VLSI Design, Third Quarter 1981, pp. 20-28.

[8] K.K. Saluja, "A Testable Design of Sequential Machines", Digest 1978, *Int. Symposium of Fault Tolerant Computing,* pp. 185-190, Toulouse, France.

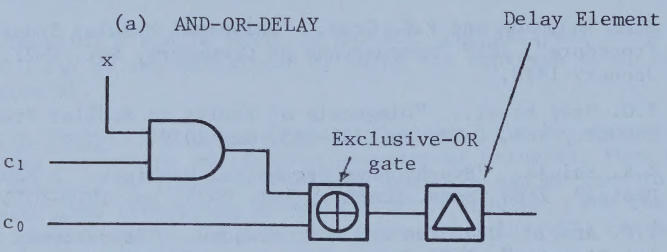
[9] S. Ramos and A.R. Smith III, "Fault Detection in Uniform Modular Realization of Sequential Machines", Digest 1972 *International Symposium on Fault Tolerant Computing*, pp. 114-119.

[10] B.A. Prasad and F.G. Gray, "Fault Diagnosis in Uniform Modular Realization of Sequential Machines", Digest 1973 *International Symposium on Fault Tolerant Computing*, pp. 157-162.

[11] Kautz, W.H. "Fault Testing and Diagnosis in Combinational Digital Circuits", *IEEE Trans. on Electron. Comput.,* Vol. EC-17, pp. 352-366, April 1962.



(a) AND-OR-DELAY


Delay Element
x
c1
c0
Exclusive-OR gate

(b) AND-EOR-DELAY

Figure 1: Basic Modules

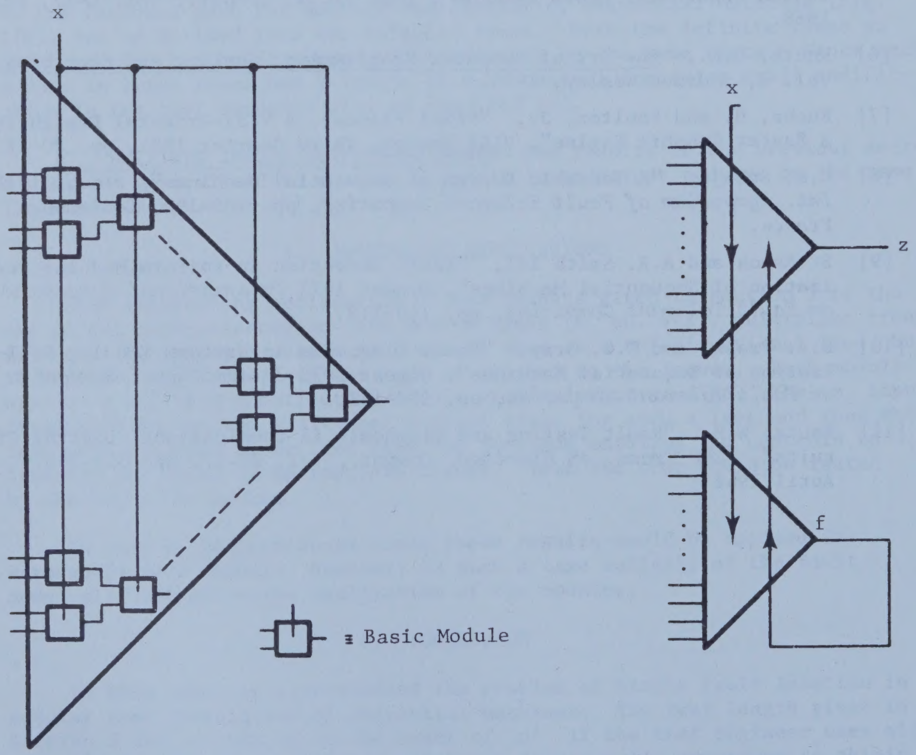

x
≡ Basic Module
x
z
f

Figure 2: (a) Tree Structure of a Definite Machine (b) Tree Realization of a Sequential Machine

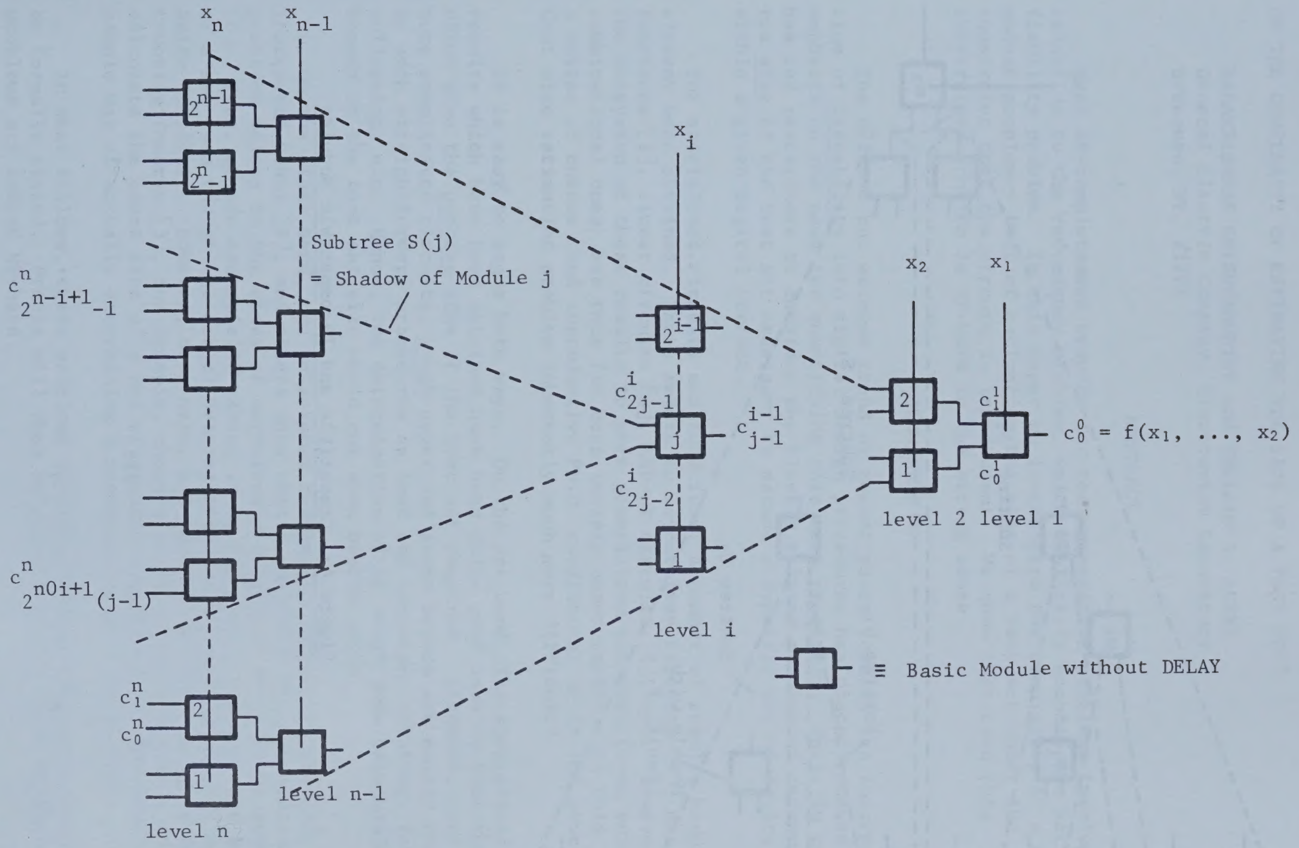

x_n
x_{n-1}
2^{n-1}
2^n-1
Subtree S(j)
= Shadow of Module j
$c^n_{2^{n-i+1}-1}$
$c^n_{2^{n0i+1}(j-1)}$
c^n_1
c^n_0
2
1
level n
level n-1
x_i
2^{i-1}
c^i_{2j-1}
j
c^{i-1}_{j-1}
c^i_{2j-2}
1
level i
x_2
x_1
2
1
c^1_1
c^1_0
$c^0_0 = f(x_1, \ldots, x_2)$
level 2 level 1
≡ Basic Module without DELAY

Figure 3: Combinational Tree

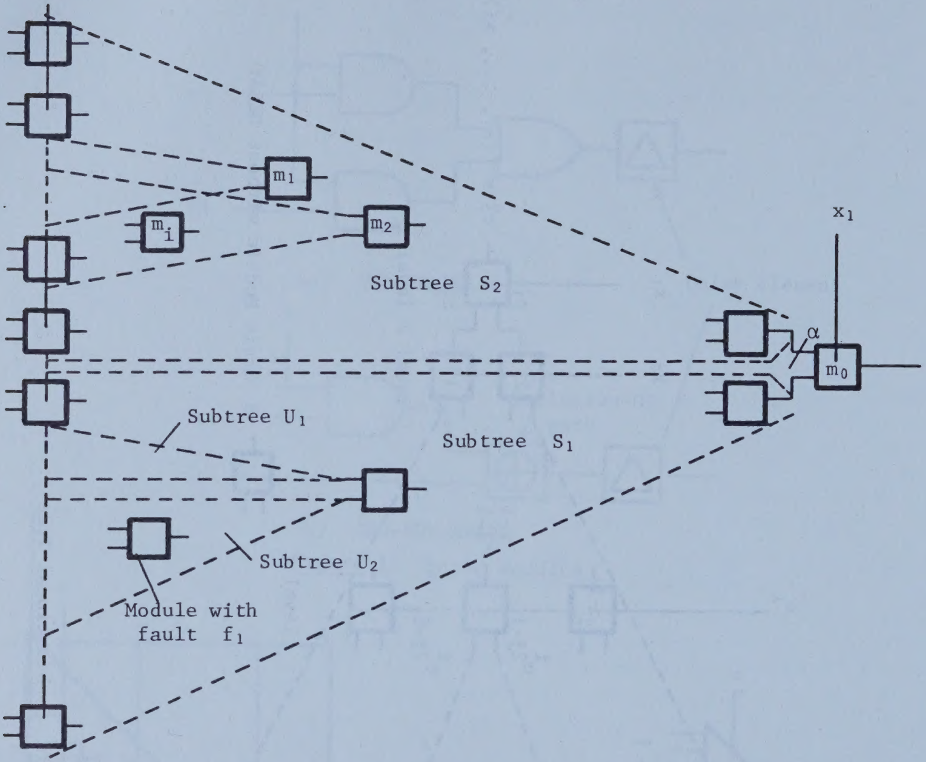

m1
m2
mi
x1
Subtree S2
α
m0
Subtree U1
Subtree S1
Subtree U2
Module with fault f1

Figure 4: Compatible and Incompatible Modules

ON THE COMPLEXITY OF ESTIMATING THE SIZE OF A TEST SET*

BALAKRISHNAN KRISHNAMURTHY and SHELDON B. AKERS
General Electric Company, Electronics Laboratory
Syracuse, NY, 13221

ABSTRACT

Most NP-completeness results for test generation problems involve a reduction to the redundancy problem, which explicitly encodes the satisfiability problem. In this paper we investigate the complexity of a more modest problem — that of estimating the size of a test set under the constraint that the circuit is irredundant. We show that even this constrained problem is NP-hard in the strong sense.

INTRODUCTION

The belated but welcome trend of recent years towards the incorporation of testability into digital design procedures has placed renewed emphasis on the need for quantifying this term precisely. This, in turn, has led researchers to consider the closely related problem of determining the size of the test set necessary to detect a specific set of faults within a given digital circuit.

For specialized circuits and functions, a number of results have already been obtained. These relate to fanout-free circuits [7], unate functions [2], linear circuits [4], AND-OR circuits [1], etc. However, the extension of these results to more generalized circuits (even purely combinational ones) has thus far been largely unsuccessful. Is this just a matter of chance (and unproductive Ph.D. candidates) or is the general test size estimation problem inherently much more difficult?

It is easy to argue both ways. On the one hand, the specialized results which have been obtained have been quite good ones in that they often give the precise size of the test set required. Likewise, even for more complicated circuits, rough upper and lower bounds are easily obtained by such straightforward procedures as lead and component counting, fault collapsing, etc. Hence, the determination of at least some reasonable bounds on the test set size would not seem unobtainable.

On the other hand, the well-known results by Ibaraki et al. [8], Ibarra and Sahni [9], and others show that a number of test generation problems belong to the class of notoriously difficult NP-complete problems [6]. Thus, we are essentially asking if we can easily estimate the size of a set even though the actual determination of the set itself may be quite difficult. However, even here, all may not be lost. Polya's famous Counting Theorem [3], for example, shows that it is often possible to calculate the exact size of a set of classes even though we may have no simple way of actually determining a representative of each class.

In what follows, three problems relating to test size estimation will be formally stated. Proofs will then be given that all three of these problems are indeed NP-hard.

---

* This work was supported by the United States Air Force, AFSC Aeronautical Systems Division under Contract No. F33615-80-C-1083.

PRELIMINARIES

In this paper we will be concerned with the problem of estimating the size of a test set to detect all single stuck-at faults in an irredundant combinational circuit. Specifically, let C be a combinational circuit. A lead ℓ in C is said to be irredundant whenever replacing ℓ by a constant logical value (0 or 1) changes the function determined by C. Equivalently, if ℓ is irredundant, then there exist appropriate tests that will detect each of the two faults: ℓ s-a-0 and ℓ s-a-1. A circuit C will be said to be irredundant if every lead in C is irredundant. Given an irredundant combinational circuit C and a subset L of the leads in C, $S^L(C)$ is defined as the minimum number of tests required to detect all single stuck-at faults on the leads in L. If L is the set of all leads in C, then we denote $S^L(C)$ by $S^*(C)$. We will investigate the following three problems associated with estimating $S^L(C)$ and $S^*(C)$:

1. Generalized Minimum Test Size (GMTS) Problem: Given an irredundant circuit C and any subset L of the leads in C, compute $S^L(C)$.
2. Minimum Test Size (MTS) Problem: Given an irredundant circuit C, compute $S^*(C)$.
3. α-Approximate Minimum Test Size (AMTS(α)) Problem: Given an irredundant circuit C and $\alpha > 1$, compute a number t such that $S^*(C) \leq t \leq \alpha \cdot S^*(C)$.

It will be shown that all of these problems are NP-hard. We do this by showing that the Satisfiability (SAT) problem is polynomially reducible to each of the above problems. To this end, we begin with a Boolean formula F and construct a circuit for which the corresponding answers will enable us to quickly determine if F is satisfiable.

First, we construct a circuit $C_1(F)$ from a Boolean formula F as shown in Figure 1. Without loss of generality, it is assumed that F is expressed in a product-of-sums form with n variables and k terms in the product. We note that the circuit realizes the formula F, i.e., for any given input, the output of the circuit will be the value of the formula. In Figure 1, the variables of F are represented as $v_1, v_2, \cdots, v_n$. There is an or-gate G_i for each of the k terms in the product. The inputs to each such gate are either the variables v_j or their complements. These inputs will be referred to as literals to indicate that they are variables with possibly complementation. We can assume that none of the G_i gates contain both

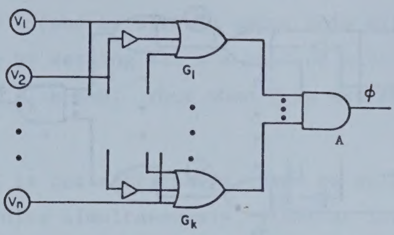


FIGURE 1 Circuit $C_1(F)$

literals of a single variable. Thus, each G_i has at most n inputs.

A primary input is an input to a gate in the circuit that is not an output of any other gate in the circuit. For example, all inputs to the G gates in $C_1(F)$ are primary inputs, whereas the inputs to the A gate are not primary inputs. Clearly, a primary input can always be set to any desired logical value. In addition, we make two simple observations on $C_1(F)$ that will be used repeatedly in the subsequent arguments:

1. The output of any one G-gate can be set to any desired logical value by appropriately setting the primary inputs to that gate.
2. The outputs of all the G-gates can be simultaneously set to 1 iff the formula F is satisfiable.

$C_1(F)$ is the first of the circuits that we will construct in this paper. Observe that $C_1(F)$ may not be irredundant. In fact, a s-a-0 fault can be detected on output lead ϕ iff F is satisfiable. This follows immediately from observation 2 and thus proves that the redundant lead problem is NP-hard — a result already shown in [9]. In the remainder of this paper, we construct irredundant circuits from arbitrary Boolean formulas and use them to prove that SAT is polynomially reducible to the three problems mentioned above. We denote polynomial reducibility by $\propto$.

MAIN RESULTS

We begin by removing the redundancy in $C_1(F)$. To do this, we construct $C_2(F)$ as shown in Figure 2. The k additional primary input leads are $w_1, w_2, \cdots, w_k$. Now the first lemma follows:

Lemma 1: For any Boolean formula F, $C_2(F)$ is irredundant.

Proof: To show that all leads of $C_2(F)$ are irredundant, it will suffice to show that any arbitrary primary input lead can be "sensitized" — i.e., values can be assigned to the other variables so that the value on this lead will appear at ϕ.

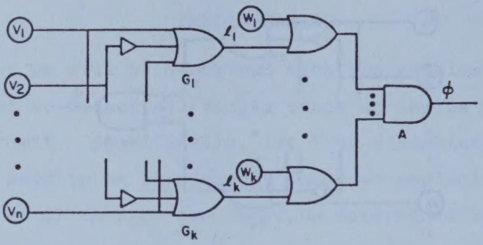


FIGURE 2 Circuit $C_2(F)$

Consider the first input lead to gate G_1 with input value a_1. To make ϕ sensitive to a_1, we set the other inputs to G_1 to 0, as well as w_1. The remaining w_i's are set to 1. Now it is easily seen that the value a_1 will appear at ϕ. (Note that this also confirms the irredundancy of all leads in the path from G_1 to ϕ.)

To sensitize a w_i input lead, it is sufficient to set all inputs to G_i to 0 and the remaining w's to 1. Since all primary inputs are either a_i or w_i inputs and all other leads are in their paths to ϕ, this confirms that all leads in $C_2(F)$ are irredundant. Q.E.D.

Theorem 1: GMTS is NP-hard.

Proof: We need to show that SAT $\propto$ GMTS. Consider an arbitrary Boolean formula F and construct $C_2(F)$ as described above. (Note that the time needed to construct $C_2(F)$ is at most a polynomial in the length of F, and further that $C_2(F)$ is irredundant, by Lemma 1.) Let L be a specific subset of the leads of $C_2(F)$; namely, the output leads $\ell_1, \ell_2, \cdots, \ell_k$ of the k G-gates. If GMTS is solvable in polynomial time, we can compute the test size, $S^L(C_2(F))$, in similar time. We shall now show that we could then use the value of $S^L(C_2(F))$ to quickly determine whether the Boolean formula F is satisfiable.

Consider lead ℓ_1. In order to detect a s-a-1 fault on ℓ_1, it will be necessary to set the inputs to G_1 to 0, as well as w_1, thus forcing a 0 on the first input on A. Likewise, all other inputs to A will have to be at 1. Now we note that no other fault on the other ℓ_i's would be detected by this test. (ϕ is normally at 0, so that to be detected a fault would have to force ϕ to 1. Clearly, only ℓ_1 s-a-1 can do this.) Thus, we conclude that each ℓ_i s-a-1 will require a separate test giving a total of k tests for the s-a-1 faults.

Now let us consider the s-a-0 faults. Many of these faults can be detected by the same test. In particular, if we can simultaneously force all of the G-gates to 1 (and as already noted this will be possible iff F is satisfiable), then by setting all w's to 0 we will have a single test which will detect any ℓ_i s-a-0. Thus when F is satisfiable, $S^L(C_2(F)) = k+1$.

Conversely, if F is not satisfiable, then we will not be able to test for all s-a-0 faults simultaneously, since at least one ℓ_i will be at 0 for any input. Hence at least two s-a-0 tests will be required and $S^L(C_2(F)) > k+1$.

Thus, F is satisfiable iff $S^L(C_2(F)) = k+1$. This shows that SAT $\propto$ GMTS and, therefore, that GMTS is NP-hard. Q.E.D.

Next we shall show that the problem of determining the minimum test size is NP-hard even if we restrict our attention to a specific set of leads — namely, the set of all leads. First, we need to develop some intermediate results.

Given an arbitrary Boolean formula F, consider the circuit $C_3(F)$ defined using $C_2(F)$. (See Figure 3.) The following two lemmas state certain properties of $C_3(F)$.

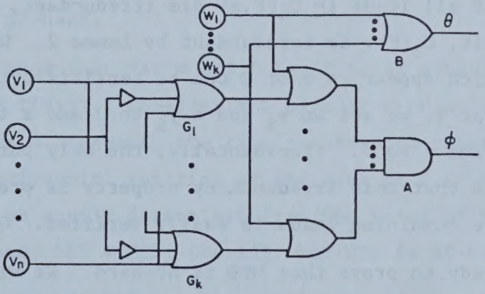


FIGURE 3 Circuit $C_3(F)$

Lemma 2: For any arbitrary Boolean formula F, $C_3(F)$ is irredundant.

Proof: In light of Lemma 1, it suffices to show that the stuck-at faults at the input and output leads of the B-gate are detectable. This is achieved by observing that the inputs to the B-gate are primary input leads and, hence, are easily sensitized. Q.E.D.

Lemma 3: The θ and ϕ outputs of $C_3(F)$ can be simultaneously set to 0 and 1, respectively, iff F is satisfiable.

Proof: This follows quite simply from the fact that to force θ to 0 requires all w_i's at 0 which, in turn, requires that all G_i outputs be 1 in order that ϕ be 1. However, as we have seen, all G_i's can be at 1 iff F is satisfiable. Q.E.D.

Now we construct a new circuit $C_4(F,m)$ defined using $C_3(F)$ and a positive integer m. This is shown in Figure 4. The primary input leads of $C_4(F,m)$ include all those of $C_3(F)$ together with 2m+1 additional primary inputs — $x_1,\cdots,x_m$, $y_1,\cdots,y_m$ and z. The following lemmas state certain properties of $C_4(F,m)$.

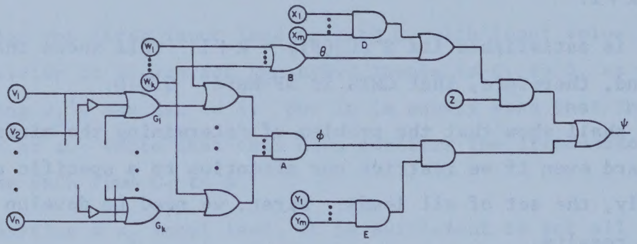


FIGURE 4 Circuit $C_4(F,m)$

Lemma 4: For any Boolean formula F and positive integer m, $C_4(F,m)$ is irredundant.

Proof: To see that all leads in $C_4(F,m)$ are irredundant, note first that the previous circuit, $C_3(F)$, is irredundant by Lemma 2. We will now show that the values which appear on θ or ϕ can be sensitized to Ψ. To make the value at θ appear at Ψ, we set an x_i and a y_i to 0 and z to 1. For ϕ, we set all y_i's to 1 and z to 0. (Incidentally, the only purpose of the z-gate is to insure that this irredundancy property is present.) The irredundancy of the remaining leads is easily verified. Q.E.D.

Now we are ready to prove that MTS is NP-hard. We will use $C_4(F,m)$ and our argument will focus on the m-input and-gates, D and E. We will show that $S^*(C_4(F,m))$ will be primarily dependent on m. The crux of the proof is Lemma 5 wherein it will be shown that to test the input leads of gates D and E, roughly m tests are required when F is satisfiable, while 2m tests will be needed if F is not satisfiable.

Counting the leads in $C_4(F,m)$ reveals that there are at most k(n+4)+2m+9 leads. (The exact number may be less since all of the G-gates may not have n inputs.) Thus, if L is the set of all leads of $C_4(F,m)$ except the input leads of gates D and E, then

$$S^L(C_4(F,m)) \le 2(k(n+4)+9)$$

(Actually, $S^L(C_4(F,m))$ will be considerably less than this value, but it will suffice for our purposes.) For ease of reference, we will denote $2(k(n+4)+9)$ by $T(F)$.

Lemma 5: For any Boolean function F and positive integer m,

$$S^*(C_4(F,m)) \begin{cases} \le m+T(F) & \text{if F is satisfiable} \\ \ge 2m & \text{if F is not satisfiable} \end{cases}$$

Proof: It is well-known that to test an m-input and-gate for single stuck-ats, it is sufficient to consider only the m input s-a-1 faults, plus the output s-a-0 fault. Since the latter will be covered by one of the T(F) tests, we now consider the tests for the s-a-1 faults on the input leads of D and E. For each gate, m such tests will be required giving a total of 2m tests unless we can perform some of the tests simultaneously. How can we test for s-a-1 faults on both D and E at the same time? To observe a s-a-1 fault on D at Ψ requires $\theta = 0$ and $z = 1$. Likewise, to observe a s-a-1 fault on E, it is seen that ϕ must be at 1. Hence, in order to test for a D and an E fault simultaneously, we must have $\theta = 0$ and $\phi = 1$. But by Lemma 3, this is possible iff F is satisfiable. Thus, we can conclude that when F is satisfiable at most $m+T(F)$ tests will be required, while in the unsatisfiable case, at least 2m will be needed. Q.E.D.

Theorem 2: MTS is NP-hard.

Proof: We need to show that SAT $\propto$ MTS. Let F be an arbitrary Boolean formula and let $m = T(F)+1$. Consider the circuit $C_4(F,m)$ shown to be irredundant by Lemma 4. (Again, the time required to construct this circuit is at most a polynomial function of the size of F.) The satisfiability of F can be easily determined from the value of $S^*(C_4(F,m))$ using Lemma 5. Hence, SAT $\propto$ MTS; proving that MTS is NP-hard. Q.E.D.

The above theorem establishes the difficulty of computing the minimum test size for an arbitrary circuit. The next result shows that it is not only difficult to compute the minimum test size, but estimating it within any fixed percentage is equally difficult! Recall that AMTS(α) is the problem of estimating the minimum test size within a factor of α.

Theorem 3: For every $\alpha > 1$, AMTS(α) is NP-hard.

Proof: We will detail the proof for $1 < \alpha < 2$. We will then merely sketch the proof for $\alpha \ge 2$.

Fix α in the range $1 \le \alpha < 2$. We need to show that SAT $\propto$ AMTS(α). Let F be a Boolean formula and m be a sufficiently large integer. (Exactly how large m should be will be determined later.) Consider $C_4(F,m)$

which, as has been observed before, can be constructed in polynomial time. Further assume that $S^*(C_4(F,m))$ can be estimated within a factor of α, i.e., we can compute a number t such that $S^*(C_4(F,m)) \le t \le \alpha \cdot S^*(C_4(F,m))$.

Now, if F is satisfiable, then by Lemma 5,

$$S^*(C_4(F,m)) \le m+T(F)$$

$$\therefore \quad t \le \alpha\cdot(m+T(F))$$

On the other hand, if F is not satisfiable, then by Lemma 5,

$$S^*(C_4(F,m)) \ge 2m$$

$$\therefore \quad t \ge 2m$$

Thus, if we choose m such that

$$\alpha\cdot(m+T(F)) < 2m$$

then we can determine the satisfiability of F from the value of t — yielding the desired result: SAT $\propto$ AMTS(α).

The value of m can be expressed by the inequality

$$m > \frac{\alpha\cdot T(F)}{2-\alpha}$$

This expression also clarifies the need to restrict this argument to $\alpha < 2$.

For $2 \le \alpha < 4$, we construct a new circuit $C_5(F,m)$ as shown in Figure 5

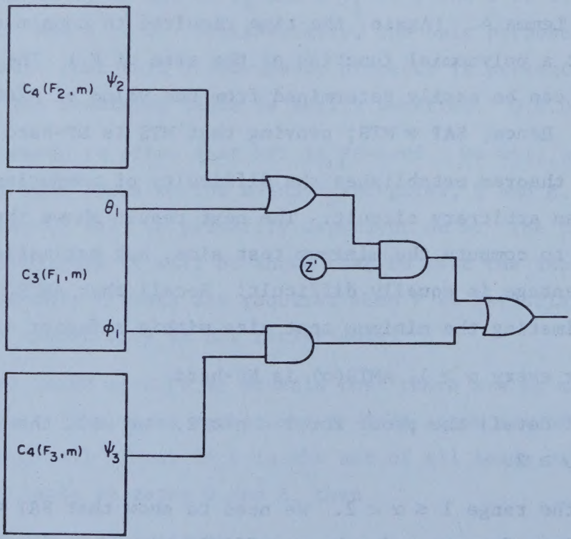


FIGURE 5 Circuit $C_5(F,m)$

— where F_1, F_2 and F_3 are Boolean formulas identical to F but with distinct Boolean variables. It can be shown that if F is satisfiable, $S^*(C_5(F,m)) \leq m+T'(F)$ for a new function $T'(F)$ that depends on F only; whereas if F is not satisfiable, $S^*(C_5(F,m)) \geq 4m$. Then by an argument very similar to the $\alpha < 2$ case, we can show SAT $\propto$ AMTS(α) for $2 \leq \alpha < 4$.

We can then carry this line of reasoning to any fixed number of steps to establish such a result for any arbitrary — but fixed — α. Hence, for every $\alpha > 1$, AMTS(α) is also NP-hard. Q.E.D.

CONCLUSION

MTS and GMTS could have been stated as decision problems that ask if k tests are sufficient to detect the specified set of faults. These decision problems are clearly in NP, for one could nondeterministically guess the k tests and verify, in polynomial time, that they indeed detect the faults of interest. Since they are in NP, we can conclude from Theorem 1 and Theorem 2 that they are, in fact, NP-complete.

Recently, Fujiwara and Toida [5] have shown that many of the NP-completeness results mentioned in [8] remain valid even for monotone functions. The NP-completeness results of this paper can also be extended to monotone functions.

The main conclusion of this paper is, of course, that the test set size estimation problem is NP-hard in the strong sense. However, several other implications are worth noting. First is the fact that all of these results hold for irredundant circuits. Thus, while redundancy has long been known to be a major complication in test generation, it is clear that it is certainly not the only contributing factor. Likewise, it should be noted that reconvergence — another nemesis of test generation — does play a key role in all of the proofs. This raises the interesting (open) question of whether or not test generation problems are more tractable for (multiple output) circuits in which both redundancy and reconvergence are precluded.

REFERENCES

[1] S. B. Akers, "Universal test sets for logic networks," IEEETC, vol. C-22, Sept. 1973, pp 835-839.

[2] R. Betancourt, "Derivation of minimum test sets for unate logical circuits," IEEETC, vol. C-20, Nov. 1971, pp 1264-1269.

[3] N. G. DeBruijn, "Polya's theory of counting," in Applied Combinatorial Mathematics (E. F. Beckenbach, Ed.), John Wiley and Sons, Inc., New York, 1964, pp 144-184.

[4] H. Fujiwara, "On closedness and test complexity of logic circuits," IEEETC, vol. C-30, Aug. 1981, pp 556-562.

[5] H. Fujiwara and S. Toida, "The complexity of fault detection problems for combinational logic circuits," IEEETC, vol. C-31, July 1982, pp 555-560.

[6] M. R. Garey and D. S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, W. H. Freeman and Company, San Francisco, 1979.

[7] J. P. Hayes and A. D. Friedman, "Test point placement to simplify fault detection," Proc. 1973 Int'l Fault Tolerant Comp. Symp., June 1973, pp 73-78.

[8] T. Ibaraki, T. Kameda and S. Toida, "NP-complete diagnosis problems on system graph," Trans. Inst. Electron. and Commun. Eng., Japan, Section E, No. 2, Feb. 1979, pp 81-88.

[9] O. H. Ibarra and S. K. Sahni, "Polynomially complete fault detection problems," IEEETC, vol. C-24, Mar. 1975, pp 242-249.

GENERALIZED DATA COMPRESSION FOR VLSI TESTING

KEWAL K. SALUJA
C.S. VENKATRAMAN
Department of Electrical and Computer Engineering
University of Newcastle,
New South Wales, 2308. Australia

1. INTRODUCTION

With the introduction of VLSI, the circuit complexity and therefore testing complexity has grown exponentially. A number of test data compression techniques have been studied in literature. The most widely analysed methods are

1. Signature Analysis [1, 5]
2. Syndrome Testing [2]
3. Transition Count Testing [3]

All the above methods have a similarity i.e. they rely on one single signature of a circuit under test. The methods of (1) and (2) are well suited to Built-In Testing (BIT) [4], and have been studied in literature, whereas (3) has only been analysed with test generation. (2) has been extended to testable design but is applicable only to combinational logic circuits, (3) has been extended to include synchronous machine testing [7]. Agarwal and Cerny [6] have proposed a generalized structure of hardware where most of the above schemes can be applied. All of these methods can be classified as follows:

(a) Weighted Data Compression
(b) Unweighted Data Compression.

This classificiation encompasses all the data compression techniques and also the conventional testing methods.

2. ALGORITHM

In this paper we propose a new scheme for testing VLSI circuits. Our scheme assumes an underlying Buil-In Logic Block Observer structure [4] and LSSD design of [8]. The generalized algorithm for testing is:

```
Step 1: Set i = 0; set a Flag
Step 2: Scan in the $i^{th}$ pattern
Step 3: Run Linear Feedback Shift Registers (LFSRs) of Built-In
        Logic Block observers, embedded in the circuit, for time $t_i$.
Step 4: Collect signature
Step 5: If testing is not complete set i = i+1, else clear flag
        and execute Step 7.
Step 6: Scan out while scanning in $i^{th}$ pattern then execute from
        Step 3.
Step 7: Scan out.
Step 8: Compare the scanned out signature with fault free signatures.
```

In our scheme the test data compression is achieved by using the Built-In Testing structure, and the flexibility of testing is maintained by using LSSD structure for the circuit. The diagnostic resolution and fault coverage is increased by dividing a single signature into multiple signatures of a desired multiplicity. Figures 1 and 2 give intuitive analysis of such testing schemes. The basic advantages of our scheme are:

(a) Small size of off-chip test storage
(b) Improved fault coverage and fault resolution. It can be demonstrated that certain faults are masked if multiple input observer blocks are used.
(c) Constrained optimization of time for testing is possible using the methods proposed in [10].

The disadvantages, as compared to BIT scheme, being multiple signatures warrant more comparison and test data size is increased, though compared to conventional testing the test data size is very small.

Our proposed scheme is being applied to certain special VLSI circuits for further analysis of time/area trade-offs.

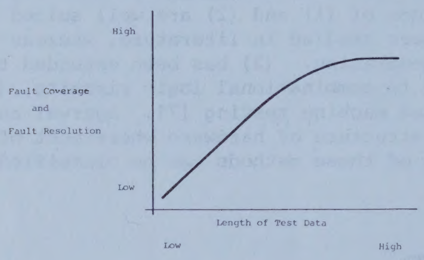


Figure 1: Relation between Data Compression and Fault Coverage

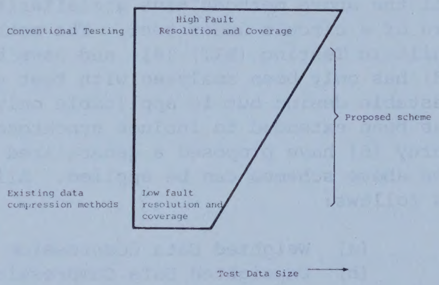


Figure 2: Comparison of Existing and Proposed Data Compression Schemes

REFERENCES

[1] R.A. Frohwerk, "Signature Analysis: A New Digital Field Service Method", *Hewlett-Packard Journal*, May 1977, pp. 2-8.
[2] J. Savir, "Syndrone-Testable Design of Combinational Circuits", *IEEE Trans. Comp.*, Vol. C-29, June 1980, pp. 442-451.
[3] J.P. Hayes, "Transition Count Testing of Combinational Logic Circuits" *IEEE Trans. Comp.*, Vol. C-25, June 1976, pp. 613-620.
[4] B. Konemann, J. Mucha, G. Zwiehoff, "Built-In Test for Complex Digital Integrated Circuits", *IEEE Jnl Solid State Circ.* SC-15, June 1980, pp. 315-319.
[5] J.E. Smith, "Measures of the Effectiveness of Fault Signature Analysis" *Proc. 11th Int. Symp. on Fault Tolerant Computing,* June 1980, pp. 35-40.
[7] C.S. Venkatraman and K.K. Saluja, "Transition Count Testing of Sequential Machines", *Proc. 10th Int. Symp. on Fault Tolerant Computing,* October 1980, pp. 167-172.
[8] E.B. Eichelberger and T.W. Williams, "A Logic Design Structure for LSI Testability", *Proc. 14th Design Auto. Conf.,* June 1977, pp. 462-468.
[9] C.S. Venkatraman, "Generalized Check-Sum Testing for Logic Networks", Ph.D. Thesis, University of Newcastle, Australia.
[10] C.R. Kime and K.K. Saluja, "Test Scheduling in Testable VLSI Circuits", *Proc. 12th Int. Symp. on Fault Tolerant Comp.,* June 1982.

ADAPTIVE DIAGNOSIS: A NEW THEORY OF t-FAULT-DIAGNOSABLE SYSTEMS

S. LOUIS HAKIMI
Department of Electrical Engineering and Computer Science
Northwestern University, Evanston, Illinois

KAZUO NAKAJIMA
Department of Electrical Engineering and Computer Science
Texas Tech University, Lubbock, Texas

ABSTRACT

In the theory of t-fault-diagnosable systems, one first chooses a set of diagnostic tests, then seeks the results of these tests, and finally proceeds to use the test results to identify the faulty units assuming that the number of faulty units does not exceed t. Nakajima was the first to suggest a departure from this practice. He proposed to adaptively choose the tests and to seek their results until one can identify a fault-free unit. This fault-free unit may then be used as a tester to identify all faulty units. In this paper, we exploit this idea fully and show that one needs the results of at most (n + 2t - 2) adaptive tests to identify all faulty units in a t-fault-diagnosable system with n units. The impact of the applications of this idea to the various models and diagnosis algorithms is examined.

1. INTRODUCTION

Theory of system level fault diagnosis has advanced very rapidly in recent years [1-18]. The effort in this area was motivated by the potential applications of this theory for automatic fault diagnosis in large electronic systems (e.g., computer networks) and was incited by the elegance and the generality of some of the main results of the theory. Before we proceed, we begin with the description of the classical digraph model.

Consider a system S of n units (subsystems, modules, chips) in which tests are conducted for the purpose of fault diagnosis. More precisely, let $U = \{u_1, u_2, \ldots, u_n\}$ represent the set of units in S and $T = \{t_1, t_2, \ldots, t_z\}$ be the set of tests to be conducted. A test $t_k \in T$ is carried out by one unit, say $u_i \in U$, on another unit $u_j \in U$. We may denote t_k by the ordered pair (u_i, u_j). Thus the units may be represented as vertices and the tests as edges of a digraph G(U,T). The result of test $t_k = (u_i, u_j)$, represented as the weight $w(t_k) = w(u_i, u_j)$ (respectively, 0 or 1) on the edge t_k of G, denotes the outcome (respectively, fault-free or faulty) of the evaluation of unit u_j by unit u_i.

S.L. Hakimi's work is supported by NSF Grants ECS-8201387 and ECS-8121741. K. Nakajima's work is supported by the Joint Services Electronics Program under ONR Contract N00014-76-C-1136.

Preparata, Metze and Chien [1] proposed the above model with the stipulation that a fault-free unit's evaluation of other units is always correct, while a faulty unit's evaluation of other units may or may not be correct, regardless of the condition of the tested unit. This assumption about the validity of the test results is referred to as the symmetric (test invalidation) digraph model. These authors defined a system S represented by digraph G(U,T) to be t-diagnosable if, given the results of all tests (or the syndrome), one can identify the faulty units provided that the number of faulty units does not exceed t. Although these authors proved many interesting results about t-diagnosable systems, it remained for Hakimi and Amin [2] to characterize t-diagnosable systems. Since then there has appeared a great many papers generalizing or modifying these results (see, for example, [3-15]), and examining potential applications of this theory to such fields as fault-tolerant computing [16] and analog fault identification [17] . In particular, characterization theorems for t-diagnosable systems under various models and in some cases assciated diagnosis algorithms have been developed [14].

In all of these results, it is explicitly assumed that one first chooses a set of tests T, then seeks the results of these tests, and finally proceeds to use the test results to identify the faulty units provided that the number of faulty units does not exceed t. Nakajima [18] was the first to suggest a departure from the above practice. Assuming that we have no more than t faulty units, he proposed that one adaptively choose the tests and obtain their results until one can identify a fault-free unit. Then one can use this unit as a tester to identify all faulty units. He proceeded to show that one can identify a fault-free unit by using the results of t(t+1) tests. This implied that (n-1) + t(t+1) tests are sufficient to identify all faulty units. This in turn implied an $O(n+t^2)$ algorithm for fault identification in certain t-diagnosable systems even without an adaptive strategy, while an O(nt) algorithm for such a purpose was previously assumed to be optimal [14].

In this paper, we will further explore this idea. In particular, in Section 2 we consider the symmetric digraph model and we propose an adaptive algorithm which identifies a fault-free unit after the application of at most 2t-1 tests, provided the number of faulty units does not exceed t. This result immediately implies that one needs at most n + 2t - 2 tests to identify all faulty units. In Section 3, we consider the asymmetric (test invalidation) digraph model. In the asymmetric digraph model a faulty unit cannot evaluate another faulty unit to be fault-free, while the remaining assumptions about the validity of tests are the same as the symmetric digraph model. We then present an optimum adaptive algorithm for the asymmetric digraph model.

2. AN ADAPTIVE DIAGNOSIS ALGORITHM FOR SYMMETRIC DIGRAPH MODEL

We begin by reminding the reader of the basic test validity assumption in this model. If $w(u_i,u_j) = 0$, then it is not possible that u_i is fault-free while u_j is faulty; while if $w(u_i,u_j) = 1$, then it is not possible that u_i and u_j are both fault-free. Given a test result, no other possibility can be excluded. We will assume that the number of units in the system S is n and that the number of faulty units in S is no greater than t. It is well known that for such a system to be t-diagnosable, $n \geq 2t + 1$, [1,2].

THE ALGORITHM FOR IDENTIFICATION OF ONE FAULT-FREE UNIT

(0) (Initialization). U is the set of units, $V_1 \leftarrow \phi$, $|V_1| \leftarrow 0$, $L \leftarrow \phi$, $|L| \leftarrow 0$, $E \leftarrow \phi$, and t is a given positive integer.

(1) If $|L| = 0$, select an arbitrary member $u_1 \in U$, place u_1 at the end of list L, set $|L| \leftarrow |L| + 1$, $U \leftarrow U - \{u_1\}$.

(2) If $|L| + 1/2|V_1| < t + 1$, let u be the last element of L and $v \in U$. Seek the result of the test $t_k = (v,u)$, set $U \leftarrow U - \{v\}$, and $E \leftarrow E \cup \{(v,u)\}$.

- (a) If $w(v,u) = 0$, place v at the end of L and set $|L| \leftarrow |L| + 1$ and return to (2).
- (b) If $w(v,u) = 1$, remove u from the end of L and set $|L| \leftarrow |L| - 1$ and set $V_1 \leftarrow V_1 \cup \{u,v\}$ and $|V_1| \leftarrow |V_1| + 2$ and return to(1).

(3) If $|L| + 1/2|V_1| = t + 1$, stop; the first element of the list L is fault-free.

Remarks: In the process of identifying a fault-free unit, the above algorithm also constructs a directed weighted forest G(V,E) where $V = L_u \cup V_1$ and L_u denotes the set of units in L. The number of roots in this forest is equal to the number of times Step (1) is used and an arbitrary member of U is selected.

Theorem 1: Let S be a system with n units of which no more than t units are faulty and $n \geq 2t+1$. Suppose the tests applied to S satisfy the assumptions of the symmetric digraph model. Then the above algorithm always identifies a fault-free unit using at most 2t - 1 tests and at most 2t + 1 units.

Proof: We first observe that at each step of the algorithm $|L| + 1/2|V_1|$ is a lower bound on the number of faulty units in S provided that the first member of L is faulty. This is because: (1) if first member u^1 of L is faulty then so are all other members of L, as for each such member $u \in L$ there is a path consisting of only zero weight edges in G(V,E) from u to u^1, (2) each pair of units placed in V_1 contains at least one faulty unit and these units are always distinct. This immediately implies that at the termination of the algorithm, when $|L| + 1/2|V_1| = t+1$, the first member of L must be fault-free.

We need to show that the number of tests is $\leq 2t-1$ and the number of units used by the algorithm is $\leq 2t+1$. Let $T_1, T_2, \ldots, T_c$ be the components of G(V,E) at the termination of the algorithm where T_c is the last component constructed. Assume $E = E^0 \cup E^1$ where E^j is the set of all edges in E with weight j, j = 0, 1. Let E_i represent the edge set of T_i and E_i^0 (respectively, E_i^1) represent those edges in E_i with weight "0" (respectively, weight "1"). It is easy to see that $|E_i^1| = |E_i^0| + 1$ for $i = 1,2,\ldots,c-1$ and $|E_c^1| = |E_c^0| - |L| + 1$. Therefore, $1/2|V_1| = |E_1| = \sum_{i=1}^{c} |E_i^1| = c - |L| + \sum_{i=1}^{c} |E_i^0| = c - |L| + |E^0|$. Thus, $|E| = |E_1| + |E^0| = c - |L| + 2|E^0|$. But from the last equality in the previous equation $|E^0| = 1/2|V_1|+|L| - c$. Thus $|E| = |L|-c+|V_1|$. However, as we know $|L|+1/2|V_1| = (t+1)$, we have $|V_1| = 2t+2-2|L|$ which in turn implies that $|E| = (2t+2)-|L|-c$ if $c \geq 2$. As $|L| \geq 1$ when the algorithm terminates, therefore $|E| \leq 2t-1$, if $c \geq 2$. Now we must examine the case when $c = 1$. We know that in this case $|E^1| = |E^0|-|L|+1$ and $|L|+|E^1| = t+1$. These equations imply that $|E| = 2t+1-|L|$. Since the algorithm never terminates with the single component unless the last test result was "0", we have $|L| \geq 2$, hence $|E| \leq 2t-1$.

To show that at most 2t + 1 units are involved in testing (or in short, to show that $|V| \leq 2t+1$), we proceed as follows. We know in a forest with c components $|E| = |V|- c$. Thus

$$|V| - c = \begin{cases} 2t + 2 - |L| - c & \text{if } c \geq 2 \\ 2t + 1 - |L| & \text{if } c = 1, \end{cases}$$

or,

$$|V| \leq \begin{cases} 2t + 1 & \text{if } c \geq 2 \\ 2t - |L| + 2 \leq 2t & \text{if } c = 1. \end{cases}$$

This completes the proof of Theorem 1.

It is clear that once a fault-free unit is identified then that unit can be used as a tester to identify the remaining faulty units. This requires at most n + 2t - 2 tests to identify all faulty units. Of course as new fault-free units are identified, they can also be used as testers to speed up this process.

One may raise a number of questions: (1) Is there an adaptive algorithm that can identify a fault-free unit in fewer tests? (2) Is there an adaptive algorithm that can identify all faulty units by a different mechanism than that of first identifying a fault-free unit? (3) If time rather than the cost of tests were the critical factor to what extent can the simultaneous (or parallel) execution of tests be used to speed up the process? (4) Suppose we are also given an initial digraph G(U,TL) (to be called test-link digraph) and we are limited to using only those tests which correspond to edges in G(U,TL), then what properties must G(U,TL) have?

We have partial answers to some of these questions: An adaptive algorithm that identifies a fault-free unit with fewer tests for almost all values of t (and never more tests than the present algorithm) has been discovered by Hakimi and Schmeichel [19]. This algorithm is somewhat more complex and has the additional feature that tests can often be executed in parallel. Although, it seems that an adaptive algorithm with ultimate goal of identifying all fault-free units need not strive to identify a fault-free unit in as few tests as possible; we have no evidence to support this intuitive statement. It can be shown that "indegree" of all vertices in G(U,TL) is $\geq$ t. This property is also a necessary condition for t-diagnosability in the "classical theory" [1,2].

It should be noted that the adaptive technique discussed above identifies all faulty unis in S (under symmetric invalidation), given that the number of faulty units does not exceed t, in n+2t-2 tests (or steps). This adaptive algorithm certainly may be applied for identifying faulty units in the classical t-diagnosable systems when all test results (nt of them) are available at the same time. In particular this can be done if the test assignment digraph for the t-diagnosable system has indegree and outdegree equal to t for all vertices. Chwa and Hakimi [16] had devised an O(nt) diagnosis algorithm for t-diagnosable system with such test assignment digraph. Our result leads to an algorithm requiring at most n+2t-2 steps for this purpose.

3. ADAPTIVE DIAGNOSIS ALGORITHM FOR ASYMMETRIC DIGRAPH MODEL

The asymmetric test invalidation digraph model was proposed and characterized by Barsi, Grandoni and Maestrini [5]. In this model, if $w(u_i,u_j) = 0$, then it is not possible that u_j is faulty; while if $w(u_i,u_j) = 1$, as before, then it is not possible that both u_i and u_j are fault-free. No other possibilities can be excluded given a test result. The motivation for this model comes from the belief that it is unlikely that, say, two faulty machines can agree on the answers to a number of problems. For a system, in this model, to be t-diagnosable, it is necessary that indegree of each vertex in the test-assignment digraph G be no less than t and that $n \geq t+2$. The potential power of this model is in its capability of identifying up to (n-2) faulty units in a system S of n units. This power was difficult to realize by constructing the appropriate test assignment digraph, but with the adaptive algorithm to be discussed here this potential can be fully exploited.

We will now describe an adaptive algorithm which identifies a fault-free unit under the asymmetric (test invalidation) digraph model. We will then proceed to prove that this algorithm is an optimum algorithm, by showing that there exists no algorithm for any value of t that can always find a fault-free unit in fewer tests.

In devising an adaptive algorithm for the asymmetric test invalidation model, one looks for the results of tests $t_i = (u_i,v_i)$ for $i = 1,2,...$ until one finds the first test, t_k, such that $w(u_k,v_k) = 0$, and this would immediately imply that v_k is a fault-free unit. Now suppose $n \geq 2t+1$, if we select u_i's and v_i's to be all distinct, then at the worst we could have $w(u_i,v_i) = 1$ for $i = 1,2...,t$ and in this case any unit $u \notin \bigcup_{\lambda=1}^{t} \{u_i,v_i\}$ is a fault-free unit and as $n \geq 2t+1$ such a unit always exists. This implies that in this case one needs at most t test to identify a fault-free unit. However, the situation becomes more complex when $n < 2t+1$. We already know that $n \geq t+2$ is a necessary condition for diagnosability. We now describe a test sequence selection for this purpose.

Consider a system S with a set of n units $\underline{U}$ and an integer t with $n \geq t+2$. Let $m = n-t-1$ and $q = \lfloor (n-1)/m \rfloor$. Let $\bar{U} = U - \{u_n\}$ with $u_n \in U$ and define $U_1,U_2,...,U_m$ to be a partition of $\bar{U}$ into m subsets whose cardinalities differ by at most one. More precisely,

$$|U_i| = \begin{cases} q + 1 & \text{for } i = 1,2,...(n-1-qm) \\ q & \text{for } i = n-qm,...,m \end{cases}$$

Let $U_i = \{u_{i1},u_{i2},...,u_{ix_i}\}$ with $q \leq x_i \leq q + 1$ for $i = 1,2,...,m$.

ALGORITHM FOR IDENTIFICATION OF ONE FAULT-FREE UNIT

(0) Initially $E \leftarrow \phi$. U_i and x_i are as defined above for $i = 1,2,...m$.

(1) Seek the result of tests $(u_{ik},u_{i\ell})$ for $i = 1,2,...,m$ and $k < \ell$ with $1 \leq k < \ell \leq x_i$. If $w(u_{ik},u_{i\ell}) = 0$ STOP, $u_{i\ell}$ is a fault-free unit.

Otherwise, set $E \leftarrow E \cup \{(u_{ik},u_{i\ell})\}$ and continue.

(2) End. u_n is a fault-free unit.

Let $G(V,E)$ be the digraph produced by the algorithm with $V = \bigcup_{i=1}^{m} U_i$. Note that if the algorithm does not terminate prematurely (i.e., no test $(u_{ik},u_{i\ell})$ with $w(u_{ik},u_{i\ell}) = 0$ is found), then the digraph has exactly m components each of which is a complete digraph (or a tournament).

Before we proceed with the proof of the validity and the optimality of this algorithm, it may be helpful to do some examples. To begin with at the worst the algorithm stops after all tests are completed and never encounters a test (u,v) for which $w(u,v) = 0$. In this case the number of tests $|E|$ is equal to $(n-1-qm)(q+1)q/2 + (m-n+qm+1)q(q-1)/2 = (2n-2-mq-m)(q/2)$. Thus the number of tests required by this algorithm $|E| \leq (2n-2-mq-m)(q/2)$.

To shed light on this upperbound on the number of tests, we consider first two extreme cases: (a) If $n \geq 2t+1$, then $q = \lfloor (n-1)/(n-t-1) \rfloor \leq 2$ with "=" occurring if and only if $n = 2t+1$. This immediately implies $|E| \leq (2n-2-mq-m)(q/2) = t$. (b) If $n = t+2$, then $m = 1$ and $q = t+1$. This immediately implies $|E| \leq (t+1)(t/2)$. Finally, let us consider a case when $t+2 < n < 2t+1$. For example, let $n = 21$ and $t = 14$; then $m = 6$ and $q = 3$. This implies $|V_i| = 4$, for $i = 1,2$; $|V_i| = 3$ for $i = 3,4,5,6$; and $|E| \leq 24$.

Theorem 2: Let S be a system of n units of which no more than t units are faulty and $n \geq t+2$. Suppose the tests applied to S satisfy the assumptions of the asymmetric digraph model. Then the above algorithm always identifies a fault-free unit using no more than $(2n-2-mq-m)(q/2)$ tests.

Proof: If $w(u_{ik},u_{i\ell}) = 0$ for some integers i,k,ℓ, then $u_{i\ell}$ is correctly identified as a fault-free unit. If, on the other hand, the algorithm does not terminate prematurely and $w(u_{ik},u_{i\ell}) = 1$ for all i,k,ℓ, then each U_i, $1 \leq i \leq m$, must contain at least $|U_i|-1 = x_i-1$ faulty units. This implies that the number of faulty units in $\bigcup_{i=1}^{m} U_i$ is at least $q(n-1-qm) + (q-1)(m-n+qm-1) = t$. This immediately implies $u_n \in U - \bigcup_{i=1}^{m} U_i$ is correctly identified as a fault-free unit. The fact that $|E| \leq (2n-2-mq-m)(q/2)$ establishes the desired upper bound on the number of tests.

We observe that this algorithm is not really an adaptive algorithm in the traditional sense. The result of a test (u,v) never effects the choice of the future tests, but merely if $w(u,v) = 0$, the algorithm terminates. Once a fault-free unit is identified, then as before this fault-free unit can be used to identify all of the remaining faulty units. This implies that in the worst case the total number of tests required to identify all faulty units $\leq (2n-2-mq-m)(q/2) + (n-1)$. It should be noted that this worst case is extremely unlikely to occur. This is because this would require the algorithm not to terminate prematurely. Also it is highly unlikely that for each i the first $|U_i|-1$ units tested by u_n are all faulty units. (One may be tempted to subtract m from the above upper bound on the total number of test; but this cannot be done for example if the result of the very last test was zero.)

It is interesting to note that many of the tests required by the algorithm may be executed in parallel. In particular, say when the test $(u_{ik},u_{i\ell})$ is being executed, any other test which does not involve either u_{ik} or $u_{i\ell}$ may also be executed. If all tests need exactly the same amount of time, say τ, then it is easy to see that, the parallel execution can quarantee that all tests are completed in $\chi'(G) \cdot \tau$ time, where $\chi'(G)$ is the edge-chromatic number of the graph G constructed by the algorithm [21]. (All edges which receive the same color can be executed simultaneously). We know that in the worst case each component of G is a complete digraph on q(or q+1) vertices. Furthermore, the edge-chromatic number of complete graph on n vertices is equal to (n-1) if n is even; and is equal to n if n is odd [21]. We can conclude that $\chi'(G) \leq (q+1)$ if q is even and $n-1-qm \geq 1$; and $\chi'(G) \leq q$, otherwise.

We now would like to prove that our algorithm for identification of a fault-free unit is optimum. More precisely, we have the following Theorem.

Theorem 3: Let n and t be any two positive integers with $n \geq t+2$. Let S be a system of n units U of which no more than t units are faulty. Suppose the tests applied to S satisfy the assumptions of the asymmetric digraph model. Then there exists no algorithm that always identifies a fault-free unit in less than $(2n-2-qm-m)(q/2)$ tests, where $m = n-t-1$ and $q = \lfloor (n-1)/m \rfloor$.

Before we prove this theorem, we need the following two lemmas. Let G(V,E) be a graph (or digraph). A subset $V_1 \subset V$ is called a vertex cover of G (respectively, independent set in G) if every edge of G has at least (respectively, at most) one end vertex in V_1. We define $\alpha_0(G)$ (respectively, $\beta_0(G)$) to be the smallest (respectively, largest) number of vertices in a vertex cover of G (respectively, independent set of G). We note that for any graph G, $\alpha_0(G)+ \beta_0(G) = n$. For $u \in V$, let G-u denote the graph obtained from G by deleting vertex u and all edges incident at u from G.

Let the weighted digraph G(U,E) represent the tests and their results performed on the units U of the system S. Assume the tests satisfy the assumptions of the asymmetric digraph model. A subset $V_1 \subset U$ is a consistent fault set (CFS) if it is consistent with the test results to assume that V_1 consists of faulty units and $U-V_1$ consists of fault-free units. It can be shown that $V_1 \subset U$ is a CFS if and only if:

(a) $(u,v) \in E$ and $v \in V_1$ imply $w(u,v) = 1$, and

(b) $(u,v) \in E$ and u and $v \in U-V_1$ imply $w(u,v)=0$.

Lemma 1: Let n, t, S and U be as in Theorem 3. Suppose the tests satisfy the assumptions of the asymmetric digraph model. The sequence of tests $t_1,\dots,t_r$ on U represented by a weighted digraph G(U,E) identifies a fault-free unit if and only if either:

(a) there exists an edge $(v,u) \in E$ with $w(v,u) = 0$; in this case, u is fault-free; or if there exists no such an edge, then

(b) there exists a unit $u \in U$ such that $\alpha_0(G-u) \geq t$; in this case, u is fault-free.

Proof: If $w(v,u) = 0$, then we know u is fault-free. Assume now that there is no weight zero edge in G, but $\alpha_0(G-u) \geq t$. As all edges have weight one, we know that every consistent fault set must contain a vertex cover of G-u. This vertex cover has cardinality $\geq t$, as a result there is no consistent fault set containing u whose cardinality $\leq t$ which implies that u is fault-free.

Now let us suppose that given the weighted digraph G(U,E), there exists a unit u which can be identified as a fault-free unit. This means there exists no consistent fault set $V \subset U$ which contains u with $V \leq t$.

A consistent fault set must contain at least one end vertex of each weight one edge and the "head" vertex of no zero weight edges. Thus, if there is an edge (v,u) with $w(v,u) = 0$ then u can be identified as fault-free. But suppose there is no zero weight edge, then each consistent fault set must contain at least one end vertex of each edge; that is, it must be a vertex cover of G. This implies that in this case for every vertex cover V^* of G which contains u, $|V^*| > t$. Or, in short $\alpha_0(G-u) \geq t$. This completes the proof of lemma 1.

We now need to state a result in extremal graph theory known as Turan's Theorem. Let m and n be two positive integers. Let $G_{m,n}$ denote a complete m-partite graph on n vertices in which all parts are as equal in size as possible. That is, the vertex set V of $G_{m,n}$ can be partitioned into m parts $V_1, V_2, \ldots, V_m$ so that their cardinalities differ by at most one such that $(u,v) \in E(G_{m,n})$ if and only if $u \in V_i$ and $v \in V_j$ for some i and j with $i \neq j$. Let $\gamma(G)$ denote the number of vertices in the largest complete subgraph of G. We now state Turan's Theorem [21].

Turan's Theorem: Let G be an n-vertex graph such that $\gamma(G) \leq m$. Then, $|E(G)| \leq |E(G_{m,n})|$; moreover, $|E(G)| = |E(G_{m,n})|$ if and only if G is isomorphic to $G_{m,n}$.

A simple consequence of Turan's Theorem may be stated as the following lemma.

Lemma 2: Let G_1 be a graph on n vertices such that $\beta_0(G_1) \leq m$. Then $|E(G_1)| \geq (n(n-1))/2 - |E(G_{m,n})|$; moreover, $|E(G_1)| = (n(n-1))/2 - |E(G_{m,n})|$ if and only if G_1 is isomorphic to the complement of $G_{m.n}$.

Proof of Lemma 2 immediately follows from Turan's theorem by letting G_1 be the compliment of G. Then $|E(G_1)| = (n(n-1)/2 - |E(G)|$ and $\gamma(G) \leq m$ if and only if $\beta_0(G_1) \leq m$.

It can be easily shown that the number of edges in the compliment of $G_{m,n}$ is equal to $(n(n-1)/2) - |E(G_{m,n})| = (2n-qm-m)(q/2)$ where $q = \lfloor n/m \rfloor$.

Proof of Theorem 3: Suppose there exists an algorithm A which adaptively chooses and seeks the results of the tests $t_1, t_2, \ldots, t_\ell$ which leads to a weighted digraph $G(U,E)$ and, of course, to the identification of a fault-free unit with $\ell < (2n-2-qm-m)(q/2)$. We first note that we need only consider the case when the results of the tests are all "one". This is because at no stage can an adaptive algorithm assuredly select a test whose result will be zero, unless the previous test results have already isolated all faulty units to within a subset $V_1 \subset U$ with $|U| \geq |V_1|+2$. In this case, however, there would be no need to seek the result of this test as a fault-free unit can already be identified. Thus, we may assume all edges in $G(V,E)$ have weight "one", but nevertheless a fault-free unit, say, $u \in U$ can be identified. By Lemma 1, $\alpha_0(G-u) \geq t$. As G-u has (n-1) vertices and $\alpha_0(G-u) + \beta_0(G-u) = n-1$, this implies that $\beta_0(G-u) \leq n-t-1 = m$. By Lemma 2, $|E(G-u)| \geq [((n-1)(n-2))/2] - |E(G_{m,n-1})| = (2n-2-qm-m)(q/2)$ where $q = \lfloor (n-1)/m \rfloor$ which contradicts the upperbound on ℓ.

4. CONCLUSIONS

The adaptive procedures for system-level fault diagnosis discussed in this paper is likely to have a major impact on the future direction of research in this area. We have already discussed matters such as: searching for an optimal algorithm for the identification of a fault-free unit in the symmetric digraph model; developing an optimal adaptive algorithm which from the outset looks for all faulty units; and studying of the cost of diagnosis, that is, is this cost more closely related to the time required to do the test or the number of tests? and what effects would such a choice have on the development of adaptive algorithms. There are other problems that should be investigated: (1) The study of adaptive algorithms for identification of intermittent faults, or the related matter of faults which occur when the diagnostic procedure is in progress. (2) Study and development of adaptive algorithm which "on the average" require the least number of tests to identify a fault-free unit (or even better to identify all faulty units) given that the number of faulty units $\leq t$. (3) The application of adaptive diagnosis procedures to more general model, say Russell-Kime [5] or Maheshwari-Hakimi [6]. (4) Chwa and Hakimi [16], in applying the theory of system diagnosis to fault-tolerant computing, found it convenient to introduce a graph model (in contrast to digraph model discussed here). In such a model a test $t_k = (u,v)$ involves merely a comparison of responses of both u and v to series of questions: If u and v always agree, then $w(t_k) = 0$; otherwise $w(t_k) = 1$. This model seems to require an observer (as in some sense nearly always an observer is required). Procedures discussed in this paper seem to be well applicable to the graph model whether under symmetric or asymmetric test invalidation assumptions. Also the adaptive diagnosis may improve the performance of fault-tolerant schemes suggested by Chwa-Hakimi [16]. (5) Recently, the theory of t-fault-diagnosable systems have been successfully applied to fault identification in analog electronic systems [17,22]. The adaptive techniques in some sense were initiated for such applications and a deeper exploration of this possibility is one of our immediate goals.

REFERENCES

[1] F.P. Preparata, G. Metze and R.T. Chien, "On the connection assignment problem of diagnosable systems," IEEE Trans. Electron. Comput., Vol. EC-16, pp. 848-854, December, 1967.

[2] S.L. Hakimi and A.T. Amin, "Characterization of connection assignment of diagnosable systems," IEEE Trans. Comput., Vol. C-23, pp. 86-88, January, 1974.

[3] J.D. Russell and C.R. Kime, "System fault diagnosis: Closure and diagnosability with repair," IEEE Trans. Comput., Vol. C-24, pp. 1078-1089, November, 1975.

[4] J.D. Russell and C.R. Kime, "System fault diagnosis: Masking, exposure and diagnosability without repair," IEEE Trans. Comput., Vol. C-24, pp. 1155-1161, December, 1975.

[5] F. Barsi, F. Grandoni and P. Maestrini, "A Theory of diagnosability without repair," IEEE Trans. Comput., Vol. C-25, pp. 585-593, June, 1976.

[6] S.N. Maheshwari and S.L. Hakimi, "On models for diagnosable systems and probabilistic fault diagnosis," IEEE Trans. Comput., Vol. C-25, pp. 228-236, October 1980.

[7] U. Manber, "System diagnosis with repair," IEEE Trans. Comput., Vol. C-29, pp. 934-937, October, 1980.

[8] P. Ciompi and L. Simoncini, "Analysis and optimal design of self-diagnosable systems with repair," IEEE Trans. Comput., Vol. C-28, pp. 362-365, May 1979.

[9] H. Fujiwara and K. Kimoshita, "Connection assignment for probabilistically diagnosable systems," IEEE Trans. Comput., Vol. C-27, pp. 280-283, March, 1978.

[10] H. Fujiwara and K. Kinoshita, "Some existence theorems for probabilistically diagnosable systems," IEEE Trans. Comput., Vol. C-27, pp. 379-384, April, 1978.

[11] J.E. Smith, "Universal system diagnosis algorithms," IEEE Trans. Comput., Vol. C-28, pp. 374-378, May, 1979.

[12] C.S. Holt and J.E. Smith, "Diagnosis of systems with asymmetric invalidation," IEEE Trans. Comput., Vol. C-30, pp. 679-690, September, 1981.

[13] S. Mallela and G.M. Masson, "Diagnosable systems for intermittent faults," IEEE Trans. Comput., Vol. C-27, pp. 560-566, June, 1978.

[14] K.-Y. Chwa and S.L. Hakimi, "On fault identification in diagnosable systems," IEEE Trans. Comput., Vol. C-30, pp. 414-422, June 1981.

[15] J.T. Butler, "Speed-efficiency-complexity tradeoffs in universal diagnosis algorithms," IEEE Trans. Comput., Vol. C-30, pp. 590-596, August, 1981.

[16] K.-Y. Chwa and S.L. Hakimi, "Schemes for fault-tolerant computing: A Comparison of modularly redundant and t-diagnosable systems," Inform. Contr., Vol. 49, pp. 212-238, June 1981.

[17] C.-C. Wu, K. Nakajima, C.-L. Wey and R. Saeks, "Analog fault diagnosis with failure bounds," IEEE Trans. Circuits Syst., Vol. CAS-29, pp. 277-284, May 1982.

[18] K. Nakajima, "A new approach to system diagnosis," in Proc. 19th Annual Allerton Conf. Commun., Contr. and Comput., September, 1981.

[19] S.L. Hakimi and E.F. Schmeichel, "An improved adaptive algorithm for system diagnosis," in preparation.

[20] J.G. Kuhl, "Fault-diagnosis in computing networks," ECS Tech Report 80-1, Dept. of Electrical and Computer Engineering, Univ. of Iowa, Iowa City, Iowa, August, 1980.

[21] J.A. Bondy and U.S.R. Murty, Graph Theory with Applications, American Elsevier Publ. Co., New York, NY, 1976.

[22] S.L. Hakimi and K. Nakajima, "On a theory of t-fault diagnosable analog systems," submitted for publication, August, 1982.

ON A DIAGNOSTIC CONFIDENCE MEASURE FOR A CLASS OF DIGITAL SYSTEMS*

I. S. HWANG** and C. R. KIME
Department of Electrical and Computer Engineering
University of Wisconsin-Madison, Madison, WI 53706

ABSTRACT

Diagnostic confidence is a measure of the probability that a system will be diagnosable. It is an alternative measure to diagnosability that is useful in situations where some or all tests are incomplete. In this paper, the diagnostic confidence is considered for a class of systems which are t-fault diagnosable with repair (t-FDWR). Two repair policies are defined for the class of systems and it is shown that the faults can indeed be identified as specified in the repair policies. Also, results are obtained which are necessary for evaluation of the diagnostic confidence and its normalized version in the systems for which limited statistical data are available. Finally, determination of the diagnostic confidence and its normalized version for the assumed diagnosis algorithms is presented in terms of subsystem failure probabilities and test fault coverages alone.

1. Introduction

Several studies in digital system diagnosis have been concerned with diagnosability, a measure of the maximum number of subsystems that can be faulty such that a system remains diagnosable [1-5]. All of these studies except [4,5] have assumed "test completeness," i.e., each test definitely fails if the assumed subsystem alone is faulty, and thus have dealt primarily with "test invalidation," i.e., each test can falsely fail or pass if some subsystem used in the test application is faulty.

In a "test incompleteness" environment, which is very likely in real applications, an alternative measure to diagnosability is needed to indicate the diagnostic figure-of-merit of a system. Otherwise, either no diagnosability or a false diagnosability is likely to be indicated for the system. The treatment of incomplete diagnostic tests is also essential for system diagnosis theory to be more useful and applicable in practice.

In the following sections, diagnostic confidence, a measure of the probability that a system will be diagnosable, and normalized diagnostic confidence, a measure of the probability that an initially faulty system will be diagnosable, are considered for a class of t-fault diagnosable with repair (t-FDWR) systems. Two repair policies for the class of t-FDWR systems are defined and justified. The diagnostic confidence and normalized diagnostic confidence are determined in terms of subsystem failure probabilities and test fault coverages.

2. Diagnostic Model

The model to be used is the one defined in [2,3] with additional model components. The model definition that is necessary for this paper is briefly stated.

---

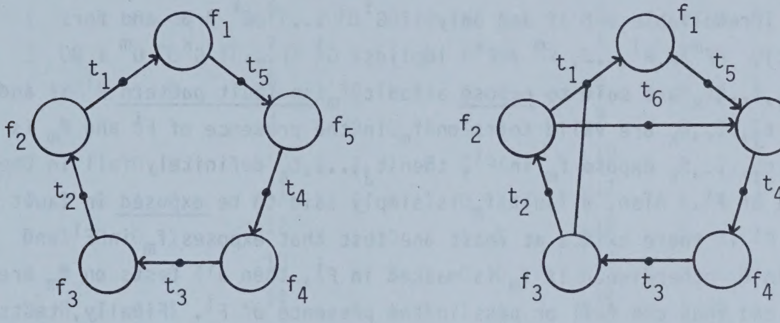
* This work was supported in part by the National Science Foundation under Grant MCS 79-09902.

** I. S. Hwang is now with the Department of Electrical Engineering, University of Minnesota, Minneapolis, MN 55455.

A system is assumed to have a set $F = \{f_1, f_2, \ldots, f_{|F|}\}$ of $|F|$ possible *faults* (potentially *faulty subsystems*) and a set $T = \{t_1, t_2, \ldots, t_{|T|}\}$ of $|T|$ pass-fail tests that can be applied to the subsystems. The faults are assumed to be permanent and their occurrences are assumed to be independent of one another. The subsystem failure probabilities (probabilities of fault occurrences) are assumed to be *a priori* available. Each test in T is applied to one and only one subsystem by a set of other subsystems, and passes if the system is fault-free. The set of tests which are applied to subsystem f_i is denoted $t^+(f_i)$ and the set of tests which are applied at least in part by subsystem f_i is denoted by $t^-(f_i)$. Each test t_j is allowed to be *incomplete* for the assumed fault in the sense that t_j does not always fail even when the assumed subsystem alone is faulty. The *fault coverage of a test* t_j is the conditional probability that t_j will fail given that the assumed subsystem alone is faulty, i.e., t_j will detect the assumed fault given t_j is valid. The fault coverage for each test is assumed to be *a priori* available. A test t_j is *valid* if each subsystem that is used in applying (performing) t_j is fault-free and *invalid* otherwise. If t_j is invalid, then t_j can fail even when the assumed subsystem is fault-free and pass even when the assumed subsystem is faulty since one or more faulty subsystems are involved in the application of t_j. Also, each test is assumed to be *consistent* in the sense that the consecutive test results remain the same if the faults which can potentially affect the test outcome are not repaired.

A *fault pattern* F^i is a set of faults that can be simultaneously present in the system and, in particular, F^0 is used to denote the empty fault pattern. For any integer t, $0 \leq t \leq |T|$, $F(t)$ denotes the collection of all fault patterns that contain t or fewer faults. A *test pattern* T^j is a set of tests that can fail simultaneously and thus is a test set result. Also, for any integer t, $0 \leq t \leq |T|$, $T(t)$ denotes the collection of all test patterns that the fault patterns in $F(t)$ can produce. An array representation of the relationship between fault patterns and test patterns is provided by the G-array. The *G-array* is $2^{|F|} \times |T|$ array having its (i,j)-entry $G^i_j = 0$, 1, or X if, for fault pattern F^i present, test t_j definitely passes, definitely fails, or possibly either passes or fails, respectively. For the present application: $G^i_j = 0$ if t_j is valid in the presence of F^i and the assumed fault is not in F^i, $G^i_j = 1$ if t_j is valid in the presence of F^i and t_j is complete for the assumed fault in F^i, and $G^i_j = X$ if t_j is invalid in the presence of F^i or incomplete for the assumed fault in F^i. A row G^i containing n entries of X represents 2^n test patterns which are all possible with F^i present in the system, in other words, G^i represents the collection of all test patterns that can occur in the presence of F^i.

Fig. 1 shows the bipartite graph representation of a single-test-per-fault (STPF) system and multiple-test-per-fault (MTPF) system [7] to be used as examples in the following sections. Fig. 2 provides the array representation of F(2) and the corresponding portion of the G-array for the MTPF system. By definition, $|t^+(f_i)| = 1$ for all f_i in F for a STPF system, and $|t^+(f_i)| > 1$ for some or all f_i in F for a MTPF system. In the array representation of F(2), $F^i_j = 0$ if f_j is not in F^i and $F^i_j = 1$ otherwise.



A) A STPF system B) A MTPF system

Fig. 1. Bipartite graphs of two systems.

3. Two Repair Policies for a Class of t-FDWR Systems

In order to develop the two repair policies for a class of t-FDWR systems that will be used in Section 4, each test in T is temporarily assumed to be complete for the assumed fault. A system is t-FDWR if and only if all faults initially present in the system are detected and identified (located) by one or more applications of the test set and repairs of identified faults provided the number of faults initially present does not exceed t. Note that the definition does not allow the repair of any fault-free subsystem [3]. The following three concepts of uniqueness/nonuniqueness of a test pattern, resolvability/irresolvability of a fault pattern, and exposure/mask of a fault are important in the development of the repair policies and significant in the establishment of the subsequent diagnostic confidence measure results.

A test pattern T^j in T(t) is unique in T(t) if T^j can occur in the presence of the one and only one fault pattern in F(t), and nonunique in T(t) otherwise. If T^j is a nonunique test pattern in T(t), then two or more fault patterns in F(t) can produce T^j. A fault pattern F^i in F(t) is resolvable in F(t) if F^i can produce only unique test patterns in T(t), and irresolvable in F(t) otherwise. If F^i is an irresolvable fault pattern in F(t), then one or more nonunique test patterns in T(t) can occur in the presence of F^i. The collection of all irresolvable fault patterns in F(t) that can produce a nonunique test pattern T^j in T(t) is the maximal irresolvable set over T^j. In general, an irresolvable fault pattern in F(t) can produce not only nonunique test patterns in T(t) but also unique test patterns in T(t), and belong to one or more maximal irresolvable sets over different nonunique test patterns in T(t). The maximal irresolvable set over a nonunique test pattern T^j in $T(t)$ is a *maximal irresolvable set* if the maximal irresolvable set over T^j is not a proper subset of the maximal irresolvable set over any other nonunique test pattern in $T(t)$ [2]. The definition of the cubical intersection $\sqcap$ is given in [8] and the resolvability/irresolvability and maximal irresolvability are restated as follows. F^i in $F(t)$ is a resolvable fault pattern in $F(t)$ if, for $F^k \epsilon F(t)$, $F^i \neq F^k$ implies $G^i \sqcap G^k = \emptyset$, and an irresolvable fault pattern in $F(t)$ otherwise. $\{F^i,\ldots,F^k\} \epsilon F(t))\}$ is a maximal irresolvable set if and only if $G^i \sqcap \ldots \sqcap G^k \neq \emptyset$ and for $F^m \epsilon F(t)$, $F^m \neq F^i,\ldots, F^m \neq F^k$ implies $G^i \sqcap \ldots \sqcap G^k \sqcap G^m = \emptyset$. Tests $t_j,\ldots,t_k$ are said to *expose* a fault f_m *in fault pattern* F^i if and only if $t_j,\ldots,t_k$ are valid tests on f_m in the presence of F^i and f_m is in F^i. If $t_j,\ldots,t_k$ expose f_m in F^i, then $t_j,\ldots,t_k$ definitely fail in the presence of F^i. Also, a fault f_m is simply said to be *exposed* in fault pattern F^i if there exists at least one test that exposes f_m in F^i and *masked* in F_i otherwise. If f_m is masked in F^i, then all tests on f_m are invalid and thus can fail or pass in the presence of F^i. Finally, tests $t_j,\ldots,t_k$ are said to *expose* a fault f_m *at step* s, $s \geqq 1$, if and only if $t_j,\ldots,t_k$ expose f_m in the fault pattern present at step s in a diagnosis-repair process.

In reference to Fig. 1 and Fig. 2, F^0, F^1, and F^9 in $F(2)$ produce unique test patterns in $T(2)$ and thus are resolvable in $F(2)$, and the other fault patterns in $F(2)$ produce nonunique test patterns in $T(2)$ and thus are irresolvable in $F(2)$. Specifically, irresolvable F^2, F^3, F^4, F^5, F^6, F^{13}, and F^{15} produce only nonunique test patterns in $T(2)$ and irresolvable F^7, F^8, F^{10}, F^{11}, F^{12}, and F^{14} produce both unique and nonunique test patterns in $T(2)$. The maximal irresolvable sets over nonunique test patterns in $T(2)$ are $\{F^2,F^6\}$, $\{F^3,F^7,f^{10}\}$, $\{F^4,F^{13}\}$, $\{F^5,F^{15}\}$ $\{F^2,F^6,F^{12}\}$, $\{F^8,F^{13}\}$, $\{F^{11},F^{13}\}$, $\{F^6,F^{12}\}$, $\{F^4,F^8,F^{11},F^{13}\}$ $\{F^{14},F^{15}\}$, and $\{F^5,F^{14},F^{15}\}$. The maximal irresolvable sets are, therefore, $\{F^2,F^6,F^{12}\}$, $\{F^3,F^7,F^{10}\}$, $\{F^4,F^8,F^{11},F^{13}\}$, and $\{F^5,F^{14},F^{15}\}$. The fault in each fault pattern in $F(1)$ is exposed in the fault pattern since the fault pattern is resolvable in $F(1)$, and all faults in F^9 are exposed in F^9 since F^9 is resolvable in $F(2)$. The fault that the fault patterns in each maximal irresolvable set have in common is exposed in each of the fault patterns, and thus the fault that the fault patterns in the maximal irresolvable set over each nonunique test pattern in $T(2)$ have in common is exposed in each of the fault patterns.

With the definitions previously given, the two repair policies are now defined for the class of t-FDWR systems. The *minimal repair policy* is to 1) upon the occurrence of a unique test pattern T^j in $T(t)$, choose for repair only the faults that are exposed in the fault pattern in $F(t)$ which

	f_1	f_2	f_3	f_4	f_5
F^0	0	0	0	0	0
F^1	1	0	0	0	0
F^2	0	1	0	0	0
F^3	0	0	1	0	0
F^4	0	0	0	1	0
F^5	0	0	0	0	1
F^6	1	1	0	0	0
F^7	1	0	1	0	0
F^8	1	0	0	1	0
F^9	1	0	0	0	1
F^{10}	0	1	1	0	0
F^{11}	0	1	0	1	0
F^{12}	0	1	0	0	1
F^{13}	0	0	1	1	0
F^{14}	0	0	1	0	1
F^{15}	0	0	0	1	1

A) F-array

	t_1	t_2	t_3	t_4	t_5	t_6
G^0	0	0	0	0	0	0
G^1	1	0	0	0	X	0
G^2	X	1	0	0	0	X
G^3	X	X	1	0	0	0
G^4	0	0	X	1	0	0
G^5	0	0	0	X	1	1
G^6	X	1	0	0	X	X
G^7	X	X	1	0	X	0
G^8	1	0	X	1	X	0
G^9	1	0	0	X	X	1
G^{10}	X	X	1	0	0	X
G^{11}	X	1	X	1	0	X
G^{12}	X	1	0	X	1	X
G^{13}	X	X	X	1	0	0
G^{14}	X	X	1	X	1	1
G^{15}	0	0	X	X	1	1

B) G-array with complete tests

	t_1	t_2	t_3	t_4	t_5	t_6
G^0	0	0	0	0	0	0
G^1	X	0	0	0	X	0
G^2	X	X	0	0	0	X
G^3	X	X	X	0	0	0
G^4	0	0	X	X	0	0
G^5	0	0	0	X	X	X
G^6	X	X	0	0	X	X
G^7	X	X	X	0	X	0
G^8	X	0	X	X	X	0
G^9	X	0	0	X	X	X
G^{10}	X	X	X	0	0	X
G^{11}	X	X	X	X	0	X
G^{12}	X	X	0	X	X	X
G^{13}	X	X	X	X	0	0
G^{14}	X	X	X	X	X	X
G^{15}	0	0	X	X	X	X

C) G-array with incomplete tests

Fig. 2. F(2)- and G(2)- arrays for the MTPF system of Fig. 1.

produces T^j, 2) upon the occurrence of a nonunique test pattern in T(t), choose for repair only the faults that are exposed in each of the fault patterns in the maximal irresolvable set over T^j, and 3) upon the occurrence of a test pattern not in T(t), choose no fault for repair. The maximal repair policy is to 1) upon the occurrence of a unique test pattern T^j in T(t), choose for repair all the faults that are in the fault pattern in F(t) which produces T^j, 2) upon the occurrence of a nonunique test pattern T^j in T(t), choose for repair all the faults that the fault patterns in the maximal irresolvable set over T^j have in common, and 3) upon the occurrence of a test pattern not in T(t), choose no fault for repair. It is noted that only faults (faulty subsystems) are repaired with both the repair policies and further that only the faults that are exposed in the fault pattern are repaired with the minimal repair policy. The following theorem which is proved in [6] by use of the three concepts defined earlier in this paper indicates that the faults can indeed be chosen for repair as specified by the two repair policies.

Theorem 1: A t-FDWR system is t-FDWR with the minimal repair policy and with the maximal repair policy.

4. Diagnostic Confidence of a Class of t-FDWR Systems

This section presents the determination of the diagnostic confidence and normalized diagnostic confidence of a t-FDWR system with the two repair policies assumed in the diagnosis-repair process. Note that each test in T is now allowed to be incomplete for the assumed fault. In the following definitions, incomplete test set T means that some or all tests in T are incomplete for the respective assumed faults. The diagnostic confidence of a t-FDWR system is the conditional probability that the system will be diagnosable, i.e., all faults initially present in the system will be detected and identified, by one or more applications of the incomplete test set and repairs of identified faults given at most t faults are initially present. The normalized diagnostic confidence of a t-FDWR system is the conditional probability that the system will be diagnosable by one or more applications of the incomplete test set and repairs of identified faults given at least one fault but at most t faults are initially present. Thus, the normalized diagnostic confidence of a t-FDWR system is the diagnostic confidence of an initially faulty t-FDWR system. The two lemmas that follow provide the conditions for a t-FDWR STPF system and for a t-FDWR MTPF system to be diagnosable with both minimal and maximal repair policies given at most t faults initially present.

Lemma 1:

1) A t-FDWR STPF system is t-FDWR with the minimal repair policy if and only if, for each of the faults initially present in the system, the test that exposes the fault at step s, $s \geqq 1$, fails at step s.

2) A t-FDWR STPF system is t-FDWR with the maximal repair policy if, for each of the faults initially present in the system, the test that exposes the fault at step s, $s \geqq 1$, fails at step s.

Proof: For sufficiency of part 1), assume that, for each of the faults initially present in the system, the test that exposes the fault at step s, $s \geqq 1$, fails at step s. Then any test pattern that occurs at step s is exactly the one that occurs when all tests in T are complete, whereby the system is t-FDWR with the minimal repair policy by Theorem 1. For necessity of part 1), note that each fault of STPF system has one and only one test on it by definition and that each of the faults initially present in the system is identified and repaired only when the test on the fault exposes the fault and the test fails. Therefore, if the test that exposes some of the initial faults at step s, $s \geqq 1$, passes at step s, the fault is never identified and repaired and thus the system is not t-FDWR with the minimal repair policy. Part 2) follows from part 1) since a t-FDWR system is t-FDWR with the maximal repair policy if the system is t-FWDR with the minimal repair policy.

Lemma 2: A t-FDWR MTPF system is t-FDWR with the minimal repair policy and with the maximal repair policy if, for each of the faults initially present in the system, all tests that expose the fault at step s, $s \geqq 1$, fail at step s.

Proof: The proof can be given in a similar way to that of Lemma 1.

The following two lemmas are used in the determination of the diagnostic confidence and normalized diagnostic confidence or their worst case bounds in situations where limited statistical information is available.

Lemma 3: If tests $t_j, \ldots, t_k$ and $t_m, \ldots, t_n$ expose faults f_g and f_h, $f_g \neq f_h$, in fault patterns F^i and F^e at step s and step s',

$s \geq s' \geq 1$, respectively, then the results of $t_j,\ldots,t_k$ at step s are conditionally independent of the results of $t_m,\ldots,t_n$ at step s' given F^i.

Proof: Assume that $t_j,\ldots,t_k$ and $t_m,\ldots,t_n$ expose f_g and f_h in F^i and F^e. Then, since $t_j,\ldots,t_k$ and $t_m,\ldots,t_n$ are valid in the presence of F^i and F^e, respectively, by definition, the results of $t_j,\ldots,t_k$ at step s and the results of $t_m,\ldots,t_n$ at step s' are determined solely by fault f_g and fault f_h, respectively, and provide no information concerning fault f_h and fault f_g, respectively. (Further, the knowledge of F^i gives the sufficient knowledge of $t_j,\ldots,t_k$ exposing f_g in F^i.) Therefore, the results of $t_j,\ldots,t_k$ at step s are conditionally independent of the results of $t_m,\ldots,t_n$ at step s' given F^i.

Lemma 4: If tests $t_j,\ldots,t_k$ and $t_j,\ldots,t_k,t_m,\ldots,t_n$ expose fault f_g in fault patterns F^i and F^h, respectively, then the joint conditional probability that $t_j,\ldots,t_k$ will fail given F^i present is greater than or equal to the joint conditional probability that $t_j,\ldots,t_k,t_m,\ldots,t_n$ will fail given F^h present.

Proof: The result is immediately claimed since the event that $t_m(,\ldots,t_n)$ fails or passes is the certain event regardless of the fault pattern present in the system.

With the aid of the foregoing lemmas, the following theorem and corollary present the determination of diagnostic confidence and normalized diagnostic confidence of the class of t-FDWR systems, respectively, with both minimal and maximal repair policies. Table 1 provides the expressions of D_{STPF}, D_{MTPF}, N_{STPF}, and N_{MTPF} in closed form.

Theorem 2:

1) The diagnostic confidence of a t-FDWR STPF system with the minimal repair policy is equal to D_{STPF}.
2) The diagnostic confidence of a t-FDWR STPF system with the maximal repair policy is greater than or equal to D_{STPF}.
3) The diagnostic confidence of a t-FDWR MTPF system with the minimal repair policy and with the maximal repair policy is greater than or equal to D_{MTPF}.

Proof: Part 1) By Lemma 1-1), the diagnostic confidence of a t-FDWR STPF system with the minimal repair policy is equal to the conditional probability that, for each of the faults initially present in the system, the test that exposes the fault at step s, $s \geq 1$, will fail at step s given that at most t faults are initially present. If F^0 is initially present in the system, the conditional probability that, for each of the faults in F^0, the test that exposes the fault at step s, $s \geq 1$, will fail at step s is 1.0 since there is no fault in F^0. Thus the probability that F^0 will be initially present in the system and for each of the faults in F^0, the test that exposes the fault at step s, $s \geq 1$, will fail at step s is $P(F^0)$. If F^i, $F^i \neq F^0$, is initially present in the system,

$$D_{STPF} = \frac{P(F^{0})^{*}+\sum P(F^{i})^{*}(\Pi P_{S}(f_{m})^{**} \text{ over all } f_{m}\epsilon F^{i}) \text{ over all } F^{i}\epsilon F(t), F^{i}\neq F^{0}}{\sum P(F^{i}) \text{ over all } F^{i}\epsilon F(t)}$$

$$D_{MTPF} = \frac{P(F^{0})+\sum P(F^{i})(\Pi P_{M}(f_{m})^{***} \text{ over all } f_{m}\epsilon F^{i}) \text{ over all } F^{i}\epsilon F(t), F^{i}\neq F^{0}}{\sum P(F^{i}) \text{ over all } F^{i}\epsilon F(t)}$$

$$N_{STPF} = \frac{\sum P(F^{i})(\Pi P_{S}(f_{m}) \text{ over all } f_{m}\epsilon F^{i}) \text{ over all } F^{i}\epsilon F(t), F^{i}\neq F^{0}}{\sum P(F^{i}) \text{ over all } F^{i}\epsilon F(t), F^{i}\neq F^{0}}$$

$$N_{MTPF} = \frac{\sum P(F^{i})(\Pi P_{M}(f_{m}) \text{ over all } f_{m}\epsilon F^{i}) \text{ over all } F^{i}\epsilon F(t), F^{i}\neq F^{0}}{\sum P(F^{i}) \text{ over all } F^{i}\epsilon F(t), F^{i}\neq F^{0}}$$

* The probability of the presence of a fault pattern is easily obtained from the subsystem failure probabilities by the assumption of independent fault occurrences.

** $P_S(f_m)$ is the fault coverage of the test on f_m.

***$P_M(f_m)$ is the joint fault coverage of all tests on f_m, i.e., the joint conditional probability that all tests on f_m will fail given the tests expose f_m.

Table 1 Expressions of D_{STPF}, D_{MTPF}, N_{STPF}, and N_{MTPF}.

the conditional probability that, for each of the faults in F^i, the test that exposes the fault at step s, $s \geq 1$, will fail at step s is $\Pi P_S(f_m)$ over all $f_m \epsilon F^i$ by Lemma 3. Thus the probability that $F^i \epsilon F(t)$, $F^i \neq F^0$, will be initially present in the system and, for each of the faults in F^i, the test that exposes the fault at step s, $s \geq 1$, will fail at step s is $P(F^i)\Pi P_S(f_m)$ over all $f_m \epsilon F^i$. Now, the claim follows noting that the probability that the fault pattern initially present in the system will be in F(t) is $\sum P(F^i)$ over all $F^i \epsilon F(t)$. Part 2) By Lemma 1-2), the diagnostic confidence of t-FDWR STPF system with the maximal repair policy is greater than or equal to the conditional probability that, for each of the faults in F^i, the test that exposes the fault at step s, $s \geq 1$, will fail at step s given that at most t faults are initially present. The remainder of proof is the same as that of part 1). Part 3) By Lemma 2, the diagnostic confidence of a t-FDWR MTPF system with the minimal repair policy and with the maximal repair policy is greater than or equal to the conditional probability that, for each of the faults initially present in the system, all tests that expose the fault at step s, $s \geq 1$, will fail at step s. Further, the conditional probability that, for each of the faults in $F^i \neq F^0$, all tests that expose the fault at step s, $s \geq 1$, will fail at step s is greater than

$D_{STPF} = 0.996 \qquad 0.995 \leqq D_{MTPF} \leqq 0.996$

$N_{STPF} = 0.893 \qquad 0.862 \leqq N_{MTPF} \leqq 0.893$

Subsystem failure probabilities used:

f_1: 0.010 f_2, f_3: 0.005 f_4, f_5: 0.008

Test fault coverages used:

t_1: 0.920 t_2, t_3, t_4: 0.900 t_5, t_6: 0.850

Table 2 Numerical Values of D_{STPF}, D_{MTPF}, N_{STPF}, and N_{MTPF}.

or equal to $\Pi P_M(f_m)$ over all $f_m \varepsilon F^i$ by Lemma 4. The remainder of proof is similar to those of parts 1) and 2).

Corollary 1:

1) The normalized diagnostic confidence of a t-FDWR STPF system with the minimal repair policy is equal to N_{STPF}.

2) The normalized diagnostic confidence of a t-FDWR STPF system with the maximal repair policy and with the maximal repair policy is greater than or equal to N_{STPF}.

3) The normalized diagnostic confidence of a t-FDWR MTPF system with the minimal repair policy and with the maximal repair policy is greater than or equal to N_{MTPF}.

Proof: The proof follows from Theorem 2 by the concept of "normalization."

Note that the fault coverage of a test on a fault is the conditional probability that the test will fail given the test exposes the fault. As seen in the above theorem and corollary, the joint conditional probability $P_M(f_m)$ that all tests on f_m will fail given the tests expose f_m is an important factor in the determination of D_{MTPF} and N_{MTPF} and thus in determination of the worst case bounds on the diagnostic confidence and normalized diagnostic confidence of a t-FDWR MTPF system. In general, since the results of tests that expose a fault f_m are conditionally dependent on one another, the exact value of $P_M(f_m)$ is not determined unless the correlations among the test results are known. The following theorem is useful in obtaining the lower and upper bounds on P_M.

Theorem 3: Let $p_j,\ldots,p_k$ be the fault coverages of n tests $t_j,\ldots,t_k$ on fault f_m. Then $\mathrm{Max}(0,\ p_j+\ldots+p_k-n+1) \leqq P_M(f_m) \leqq \mathrm{Min}(p_j,\ldots,p_k)$.

Proof: Let A, B, and C denote three events. The lower and upper bounds are obtained by repeatedly using the basic laws of a conditional probability measure $P(A \cap B|C) \geq P(A|C) + P(B|C) - 1$ and $P(A \cap B|C) \leq P(A|C)$ (or $P(B|C)$). The formal proof is given in [6].

The lower and upper bounds on P_M as given in Theorem 3 are general in the sense that no specific assumption is made in their derivation. On the other hand, the bounds, especially the lower bound, may be too conservative to be used even if the correlations of the test results are only approximately known. One further observation indicates that as the number of incomplete tests on a fault increases, the tests should be carefully designed so that the fault coverages of individual tests are high and positively correlated so that $P_M(f_m)$ remains close to 1 and D_{MTPF} and N_{MTPF} are not significantly degraded. Table 2 shows the numerical values of D_{STPF}, D_{MTPF}, N_{STPF}, and N_{MTPF} of the two systems of Fig. 1 along with the subsystem failure probabilities and test fault coverages used in the computation.

5. Conclusion

Special consideration has been given to the evaluation of diagnostic confidence and normalized diagnostic confidence for a class of t-FDWR systems in terms of only the statistical data that is usually assumed to be a priori available. In general, it is extremely difficult to obtain the conditional probabilities of individual invalid test results, and thus their use has been avoided. Also, it has been indicated that the multiple tests on a subsystem should be carefully derived. Otherwise, the joint fault coverage of multiple tests can be poor even though the fault coverages of individual tests are very high. Poor joint fault coverage can, in turn result in low diagnostic confidence.

6. References

1. F. P. Preparata, G. Metze, and R. T. Chien, "On the connection assignment problem of diagnosable systems," IEEE Tran. Electronic Computers, Vol. EC-16, pp. 848-854, December 1967.
2. C. R. Kime, "An analysis model for digital system diagnosis," IEEE Trans. on Computers, Vol. C-19, pp. 1063-1073, November 1970.
3. J. D. Russell and C. R. Kime, "System fault diagnosis: closure and diagnosability with repair," IEEE Trans. on Computers, Vol. C-24, pp. 1078-1089, November 1975.
4. M. L. Blount, "Probabilistic treatment of diagnosis in digital systems," Proc. 7th Int. Conf. on Fault-Tolerant Computing, pp. 72-77, June 1977.
5. M. L. Blount, "Modeling of diagnosis in fail-softly computer systems," Proc. 8th Int. Conf. on Fault-Tolerant Computing, pp. 53-58, June 1978.
6. I. S. Hwang. "On the confidence of the diagnosis-repair process for digital systems," Ph.D. Dissertation (in preparation), University of Wisconsin-Madison.
7. C. S. Holt and J. E. Smith, "Diagnosis of systems with asymmetric invalidation," IEEE Trans. on Computers, Vol. C-30, pp. 675-690, September 1981.
8. D. L. Dietmeyer, Logic Design of Digital Systems, 2nd Edition, Boston: Allyn and Bacon, 1978.

ORDER STATISTICS ON DISTRIBUTED SETS

NICOLA SANTORO
Distributed Computing Group
Carleton University
Ottawa, Ontario
Canada

JEFFREY B. SIDNEY
Department of Mathematics and Statistics
Carleton University
Ottawa, Ontario
Canada

ABSTRACT

We prove lower and upper bounds on the communication complexity of finding the median of a set distributed without replication at the sites of a communication network. For a fixed number of sites, these bounds are sharp with respect to order of magnitude.

1. INTRODUCTION

Given a totally ordered set $\mathcal{U}$, unbounded both above and below, let V^d be a set of d n-dimensional sorted vectors in $(\mathcal{U})^n$ with no replication or common elements. The set $U^d = \{ V_i[j] \mid V_i \in V^d, 1 \leq j \leq n\}$ of all elements stored in the vectors $V_i \in V^d$ will be called a d-distributed set, or d-set, where $V_i[j]$ denotes the j-th component of V_i. We shall denote by $m(U^d)$ the lower median of V^d; i.e. $m(U^d)$ is the $(\lceil n.d/2 \rceil)$-th smallest element in U^d. Given $V^d = \{ V_0, V_1, \ldots, V_{d-1}\}$, let $V \in V^d$ be an arbitrarily chosen vector, called the base vector; without loss of generality, let $V = V_o$.

When operating on a d-set U^d with base vector V_o, the atomic datum, called d-datum, is any (d-1)-tuple $\underline{x} = < x_1, x_2, \ldots, x_{d-1} >$ where $1 \leq x_i \leq n$, $1 \leq i \leq d-1$. Given a d-datum $\underline{x}$, let $t(\underline{x}) = < \ell_1, \ell_2, \ldots, \ell_{d-1} >$ be the (d-1)-tuple where ℓ_i is the rank of $V_i[x_i]$ in $V_{\underline{x}} = \{ V_o[\ell] \mid 1 \leq \ell \leq n\} \cup \{ V_j[x_j] \mid 1 \leq j \leq d-1 \}, 1 \leq i \leq d-1$.

The atomic operation called d-operation on a d-set U^d with base vector V_o is composed of three indivisible activities: (1) specifying a d-datum $\underline{x} = (x_1, \ldots, x_{d-1})$; (2) accessing the values $V_1[x_1], \ldots, V_{d-1}[x_{d-1}]$; (3) determining $t(\underline{x})$.

We consider the problem of efficiency determining $m(U^d)$ using d-operations. The computational complexity of the problem is defined as

$$C_n(d) = \min_{A} \max_{V^d} C_n(A, V^d)$$

where $C_n(A, V^d)$ is the number of d-operations required by algorithm A to find the median of U^d. In other words, $C_n(d)$ represents the maximum number of d-operations used by the algorithm that finds the median of a d-set the fastest in the worst case. We are interested in determining upper and lower bounds on $C_n(d)$ for all d. We assume that n, the dimension of the vectors in V^d, is much greater than d, the cardinality of V^d.

This problem originates from the following practical situation. Consider a file distributed without replication at the nodes of a computer-communication network; order-statistics queries about the median of the file can be received at any node, and each node is aware only of the data stored locally [7]. The node at which a query is originated becomes the initiator of the query-resolution algorithm. The execution of a resolution algorithm involves both processing activities (e.g. comparison, search, selection, etc.) at the nodes, and communication activitives (e.g. message exchange, data transfer, etc.) between the nodes. Since processing time is usually neglegible when compared to transmission time, the efficiency of a solution algorithm is measured solely in terms of communication activities [7]. A basic communication activity for answering a query is the process of the initiator node requesting and receiving the k-th smallest data element from each of the other nodes. It is not difficult to see that $C_n(A, V^d)$ represents the number of communication activities required by algorithm A to find the median of a file equidistributed among d nodes, where the base vector represents the data stored at the initiator node. The restrictions to all nodes having the same amount of stored data, and to finding the median rather than the k-th smallest element are made only to simplify the presentation and can be easily lifted [9].

Note that the "election" and "ranking" algorithms presented in the literature, e.g. [1-6,8], originate from problems related, to a certain degree, to the one discussed here. In particular, these algorithms can be seen as topology-concerned techniques for finding the smallest and the k-th smallest element of a d-set, respectively, in the very special case when only one element is stored at each node; i.e., n = 1.

The problem of determining $m(U^d)$ is of interest only when d > 1. It is not difficult to derive the bound $C_n(d) \leq O(n\ d\ )$. A natural question is whether the structure of the problem can be exploited to obtain a faster algorithm. In this paper, we answer this question affirmatively and show that for $d > 1$,

$$C_n(d) \leq 0\ (\log\ (nd)/\log(d/(d-1))) \tag{1.1}$$

$$C_n(d) \geq 0\ (\log\ n). \tag{1.2}$$

The above bounds are sharp for $d = 2$ with respect to the order of magnitude.

2. LOWER BOUND

Lemma 2.1. $\quad C_n(d-1) \leq C_n(d)$

Proof: Consider a set $V^{d-1} = \{V_o, \ldots, V_{d-2}\}$ of $(d-1)$ n-dimensional vectors in $(U)^n$. Let $x_1 = \min\{y \in U^d\}$ and $x_2 = \max\{y \in U^d\}$, and let $f(n,d)$ be as follows

$$f(n,d) = \begin{cases} n/2 & \text{if } n \text{ is even} \\ (n-1)/2 & \text{if both } n \text{ and } d \text{ are odd} \\ (n+1)/2 & \text{otherwise} \end{cases}$$

Define $\underline{x}$ to be the sorted n-dimensional vector with distinct elements from $U-U^{d-1}$, such that $\underline{x}(i) < x_1$ for $1 \leq i \leq f(n,d)$ and $\underline{x}(i) > x_2$ for $f(n,d) < i \leq n$, where $\underline{x}(i)$ denotes the i-th element of $\underline{x}$. Let $V^d = V^{d-1} \cup \{\underline{x}\}$; it is clear that $p = m(U^{d-1})$ if and only if $p = m(U^d)$. Hence, for finding the median of V^{d-1} it suffices to find the median of V^d. Therefore, $C_n(d-1) \leq C_n(d)$. □

Let $C_{a,b}(2)$ denote the computational complexity of the problem of finding the median of the set $\{V_o[i] | 1 \leq i \leq a\} \cup \{V_1[j] | 1 \leq j \leq b\}$ for any choice of V_o and V_1.

Lemma 2.2 [9]. $C_{n_o,n_1}(2) \geq C_{n'_o,n'_1}(2) \quad$ if $n'_j \leq n_j$.

Lemma 2.3. $C_n(2) \geq \lfloor \log n \rfloor + 1$.

Proof: By induction. The lemma trivially holds for $n \leq 4$. Assume the result holds for $n \leq 2^{k-1}$ for $k \geq 3$. Let $n = 2^k$. We now show that there is a choice of vector V_1 which makes the lemma hold. It is not difficult to see that for each 2-datum $<j>$ there exists a vector V'_1 such that

$t(<j>) = 1$, and a vector V_1'' such that $t(<j>) = n+1$. If $j \leq n/2$, choose V_1 to be V_1'; in this case, we can eliminate from consideration only the upper j elements of V_o and the lower j elements of V_1'. That is, $C_n(2) \geq 1 + C_{n-j, n-j}(2)$; by Lemma 2.2 and the inductive hypothesis, we have $C_n(2) \geq 1 + C_{(n/2)}(2) \geq 1 + k$, which proves the lemma for this case. If $j > \frac{n}{2}$, choose V_1 to be V_1''; then we can eliminate from consideration only the upper n-j+1 elements of V_1 and the lower n-j elements of V_o. Therefore, $C_n(2) \geq 1 + C_{j,j-1}(2)$; by Lemma 2.2 and the inductive hypothesis, we have $C_n(2) \geq 1 + C_{(n/2)}(2) \geq 1 + k$ which completes the proof for $n = 2^k$. Application of lemma 2.2 covers the cases of $2^{k-1} < n < 2^k$. $\square$

Therefore by Lemmas 2.1 and 2.3 we have shown the following:

Theorem 2.1 For any $d \geq 2$, $C_n(d) \geq C_n(d-1) \geq \ldots \geq C_n(2) \geq 0(\log n)$, so that $0(\log n)$ comparisons are needed for finding the median of a d-set in the worst case.

3. UPPER BOUND FOR d = 2

In this section, we shall present an algorithm which achieves $C_n(2) = 0(\log n)$. In the subsequent section we show how to generalize this result to achieve the general bound asserted in (1.1). Without loss of generality, we assume that $n = 2^r$ for some positive integer r. Given a k-dimensional vector $\underline{x} = < x_1, \ldots, x_k >$ and two positive integers p and q, let $p' = \min\{p,k\}$, $q' = \min\{q,k\}$, and let $\uparrow(\underline{x}, p, q)$ denote the α-dimensional vector $< x_1', x_2', \ldots, x_\alpha' >$ where $\alpha = k - q' + p' - 1$ and $x_j' = x_j$ if $j < p'$, $x_j' = x_{j+q'-p'+1}$ otherwise.

Algorithm 3.1

```
This algorithm finds $m(U^2)$ given $V^2 = \{V_o, V_1\}$.
1. Set $\overline{V}_o := V_o$, $\overline{V}_1 := V_1$, and $\beta := n/2$.
2. If $\overline{V}_o[\beta] < \overline{V}_1[\beta]$ set $\overline{V}_o := \uparrow(\overline{V}_o, 1, \beta)$ and $\overline{V}_1 := \uparrow(V_1, \beta+1, |V_1|)$;
   otherwise set $\overline{V}_1 := \uparrow(\overline{V}_1, 1, \beta)$ and $\overline{V}_o := \uparrow(\overline{V}_o, \beta+1, |\overline{V}_o|)$
3. Set $\beta := \beta/2$. If $\beta \geq 1$ go to step 2; otherwise, set $m(U^2) := \min$
   $\{\overline{V}_o[1], V_1[1]\}$ and halt.
```

Lemma 3.1. Algorithm 3.1 terminates.

The proof is trivial.

Let s denote the number of times step 2 is executed, let β_t, $\overline{V}_{o,t}$, $\overline{V}_{1,t}$, $\overline{U}_t$ denote the value of variable β, the vectors $\overline{V}_o$ and $\overline{V}_1$, and the set of elements of $\overline{V}_o$ and $\overline{V}_1$, respectively, before the t-th execution of step 2, $1 \leq t \leq s$.

Lemma 3.2. $m(U_t) = m(U^2)$ for $1 \leq t \leq s$.

Proof: By induction. The theorem trivially holds for $t = 1$. Let it hold for $t = T \geq 1$. Without loss of generality, let $\overline{V}_{o,t}(\beta_t) > \overline{V}_{1,t}(\beta_t)$; then $V_{1,t}(i) \neq m(U_t)$ for $1 \leq i \leq \beta_t$, and $\overline{V}_{o,t}(j) \neq m(U_t)$ for $\beta_t + 1 \leq j \leq 2\beta_t$. This implies that $\overline{V}_{o,t+1} = \uparrow(\overline{V}_{o,t}, \beta_t + 1, 2 \cdot \beta_t)$ and $\overline{V}_{1,t+1} = \uparrow(\overline{V}_{1,t}, 1, \beta_t)$ still contain $m(U_t)$. Since we have eliminated β_t elements smaller than $m(U_t)$ and β_t elements larger than $m(U_t)$, then $m(U_t) = m(U_{t+1})$. Therefore, $m(U_t) = m(U^2)$ for $1 \leq t \leq s$. □

Therefore, by Lemmas 3.1 and 3.2, we have shown the following:

Theorem 3.1. Algorithm 3.1 correctly determines $m(U^2)$.

Let us now estimate the number of 2-operations performed by Algorithm 3.1. The algorithm halts when, in step 3, the value of variable β, initially set to $n/2$, becomes smaller than one. Since β is halved after every execution of step 2, and since only one 2-operation is performed in that step, then $O(\log n)$ 2-operations are performed in total. Thus, we proved that $O(\log n)$ is an upper-bound to $C_n(d)$ for $d=2$. Together with the lower-bound presented in Section 2, we have:

Theorem 3.2. $C_n(2) = O(\log n)$.

4. GENERAL UPPER-BOUND

Let $C_n^k(d)$ denote the computational complexity of the problem of finding the k-th smallest element of a d-set using d-operations. Without loss of generality, let $k \leq \lceil n\, d/2 \rceil$.

Lemma 4.1 [9]. $C_{n-1}^k(d) \leq C_n^k(d)$.

Lemma 4.2 [9]. $C_n^n(d) \leq O(\log n/\log(d/(d-1)))$.

Theorem 4.1. $C_n(d) \leq O(\log(n \cdot d)/\log(d/(d-1)))$.

Proof: Let $s = \lceil n \cdot d/2 \rceil$. From Lemma 4.1, it follows $C_n(d) = C_n^s(d) \leq C_s^s(d)$ for $d \geq 2$. From Lemma 4.2, it follows $C_s^s(d) \leq O(\log s/\log(d/(d-1)))$. Therefore, $C_n(d) \leq O(\log(n \cdot d)/\log(d/(d-1)))$. □

5. CONCLUDING REMARKS

In this paper we have proved lower and upper bounds on the communication complexity of finding the median of a distributed set. For a fixed d, these bounds are sharp with respect to order of magnitude.

The problem of finding the k-th smallest element of a distributed set (partially discussed in section 4) is analyzed in detail in [9].

ACKNOWLEDGEMENT

The authors would like to thank Doron Rotem for his helpful suggestions. This work has been supported in part by the Natural Sciences and Engineering Research Council of Canada.

REFERENCES

[1] E.J. Chang and R. Roberts,"An improved algorithm for decentralized extrema-finding in circular configurations of processors", CACM 22, 281-283 (1979).

[2] D. Dolev, M. Klawe, and M. Rodeh, "An O(N log N) unidirectional distributed algorithm for extrema-finding in circles", J. of Algorithms, to appear.

[3] W.R. Franklyn, "On an improved algorithm for decentralized extrema finding in circular configurations of processors", CACM 25, 336-337 (1982).

[4] R.G. Gallager, "Finding a leader in a network with O(E) + O(N log N) messages", Internal Memorandum, M.I.T, 1979.

[5] D.S. Hirschberg and J.B. Sinclair, "Decentralized extrema-finding in circular configurations of processors", CACM 23, 627-628 (1980).

[6] E. Korach, D. Rotem, and N. Santoro, "Distributed algorithms for ranking the nodes of a network", Proc. 13th SE Conf. on Combinatorics, Graph Theory and Computing , to appear .

[7] B.W. Lampson, M. Paul, and H.J. Siegert (eds.), "Distributed Systems -Architecture and Implementation", Springer-Verlag, New York, 1981.

[8] N. Santoro, E. Korach, and D. Rotem, "Decentralized extrema-finding in circular configurations of processors: an improved algorithm", Proc. 11th Conf. on Numerical Mathematics and Computing , 401-412,1981.

[9] N. Santoro and J.B. Sidney, "Communication bounds for selection in distributed sets", Distributed Computing Group, Res. Rep. SCS-TR-9, Carleton University, 1982.

PARALLEL GRAPH ALGORITHMS WITHOUT MEMORY CONFLICTS

D. S. HIRSCHBERG
ICS Department
University of California at Irvine
Irvine, California 92717

ABSTRACT

The WRAM is a model of parallel computation that assumes that there is a global memory, to each cell of which all processors have immediate random access. Simultaneous reads from the same cell are permitted; simultaneous writes to the same cell are permitted only if all values to be written are the same. Several graph problems have been solved in time $(\lg n)^2$ if simultaneous writes are forbidden. We show that these problems can be solved in time lg n on the WRAM and thus can be solved in time $(\lg n)^2$ even when both simultaneous reads and writes are forbidden. The WRAM algorithms are, for most of these problems, relatively straightforward. However, the algorithm for solving the minimum spanning tree problem is a probabilistic solution.

INTRODUCTION

The shared memory model for parallel computation assumes that there is a global memory, to each cell of which all processors have immediate random access. The shared memory model has many variants including the PRAC, PRAM, and WRAM.

The PRAC (defined in [LPV] but used earlier by others) does not allow simultaneous read from or write to a single cell by more than one processor.

The PRAM (defined in [FW] but used earlier by others) allows simultaneous reads from a cell but does not allow simultaneous writes.

The WRAM (defined in [BH] so as to include the models of [SV][Go]) allows simultaneous reads and also allows simultaneous writes. Simultaneous

---

This work supported by the National Science Foundation under grant MCS-82-00362.

multiple writes to one location can be resolved in different ways. We use the definition in [SV] allowing simultaneous writes only if all values to be written are the same.

Many problems have been solved under the PRAM model of computation. Problems already solved include sorting in time O(lg n lglg n) using n processors [BH] and in time O(lg n) using n lg n processors [Pr], and several graph problems in time $O(\lg^2 n)$ including transitive closure [Ar], topological sort [DNS], connected components [CLC], bi-connected components [Sa], and minimum spanning tree [Sa].

Thus, it is known how to solve these graph problems in time $O(\lg^2 n)$ if simultaneous reads are permitted. We show that is possible to solve these problems in time $O(\lg^2 n)$ even if simultaneous reads are prohibited. We obtain these results by first exhibiting O(lg n) algorithms under the WRAM model of computation. Any algorithm taking time T(n) on a WRAM can be converted into an algorithm taking time T(n)lg n on a PRAC [Ec]. Therefore, these graph problems can be solved on a PRAC in time $O(\lg^2 n)$.

(LG N)-TIME ALGORITHMS ON A WRAM

WRAM transitive closure

```
1. for all x,y  C[x,y]  ← edge(x,y)
   for all x      C[x,x]  ← TRUE
2. for lg n iterations do step 3
3.      for all x,y,z such that C[x,z] AND C[z,y]
                  C[x,y]  ← TRUE
```

Step 3 can be done in parallel in time O(1) using n^3 processors. Thus total time is O(lg n) using n^3 processors.

WRAM all pairs shortest paths

```
1. for all i,x,y   A[i,x,y]  ← FALSE
   for all x,y     A[1,x,y]  ← edge(x,y)
   for all x       A[0,x,x]  ← TRUE
   maxk  ← 1
2. for lg n iterations do steps 3,4
```

```
3.      for all i,x,y,z such that A[maxk,x,y] AND A[i,y,z]
                                A[maxk+i,x,z]  ←  TRUE
4.      maxk  ←  2*maxk
5. for all i,j,x,y such that A[i,x,y] AND A[j,x,y]
        if i<j then             A[j,x,y]  ←  FALSE
6. for all i,x,y such that A[i,x,y]
                                shortest(x,y)  ←  i
```

Prop.: If the minimum path x → y is of length j then A[j,x,y] will be TRUE after maxk ≥ j.

Proof: By induction on the minimum path length. The basis (path length 1) is trivially true. If minimum path is of length j, let z be the node that is maxk edges from x along such a minimum path. The minimum path x → z is of length maxk (else x → y would have a shorter than j path) and A[maxk,x,z] will be TRUE by inductive hypothesis. Similarly, the minimum path z → y is of length i = j-maxk and A[i,z,y] will be TRUE. Thus, the algorithm will set A[j,x,y] to TRUE. We note that Step 3 can be done in parallel in time O(1) using n^4 processors. Thus total time is O(lg n) using n^4 processors.

WRAM all pairs longest paths

```
1. for all i,x,y    A[i,x,y]  ←  FALSE
   for all x,y      A[1,x,y]  ←  edge(x,y)
   for all x        A[0,x,x]  ←  TRUE
2. for lg n iterations do step 3
3.      for all i,j,x,y,z such that A[i,x,y] AND A[j,y,z]
                                A[i+j,x,z]  ←  TRUE
4. for all i,j,x,y such that A[i,x,y] AND A[j,x,y]
        if i>j then             A[j,x,y]  ←  FALSE
5. for all i,x,y such that A[i,x,y]
                                longest[x,y]  ←  i
6. for all x,y,z such that longest[z,z]>0
        if longest[x,z] ≥ 0 AND longest[z,y] ≥ 0
        then                    longest[x,y]  ←  infinity
```

A[i,x,y] will be TRUE iff there is a path x → y of length i.

Step 3 can be done in parallel in time O(1) using n^5 processors.

Thus total time is O(lg n) using n^5 processors.

WRAM topolgical sort   (based on [DNS])

```
1. do all pairs longest paths (produces array longest[*,*])
```

```
2. for all x        depth[x] ← max{ longest[y,x] }
3. sort { <depth[x],x> }
```

Step 1 can be done in O(lg n) time using n^5 processors.

Step 2 can be done in O(lg n) time using n^2 processors.

Step 3 can be done in O(lg n) time using n^2 processors.

Thus total time is O(lg n) using n^5 processors.

WRAM connected components

```
1. do transitive closure (produces array C[*,*])
2. for all x        component[x] = min { y | C[x,y] is TRUE }
```

Thus total time is O(lg n) using n^3 processors.

WRAM biconnected components

```
1. for all x        artic[x]  ←  FALSE
   for all x,y      samecomp[x,y]  ←  TRUE
2. for all x do steps 3,4
            (check to see if x is an articulation point)
3.      do connected components on G - {x}
        (produces array component[*], different for each x)
4.      for all y,z
            if component[y] ≠ component[z]
            then  artic[x]  ←  TRUE
                  samecomp[y,z]  ←  FALSE
5. for all x  bicomp[x]  ←  min{ y | samecomp[x,y] = TRUE }
```

Steps 3,4 and 5 are done n times in parallel.

Step 3 can be done in time O(lg n) using n^3 processors.

Step 4 can be done in time O(1) using n^2 processors.

Step 5 can be done in time O(lg n) using n^2 processors.

Thus total time is O(lg n) using n^4 processors.

WRAM minimum spanning tree

```
1. for all x        comp[x]  ←  x
2. while there exists x,y s.t. comp[x]≠comp[y] do steps 3-7
3.      for all x
            local[x]  ←  min{ edge[x,y] | comp[y]≠comp[x] }
            link[x]  ←  min{ y giving local[x] }
            if the set is empty then link[x]  ←  NULL
```

```
4.      for all x s.t. comp[x] = x
             global[x] ← min{ local[y] | comp[y] = x
                                          AND link[y] ≠ NULL }
             frompt[x] ← min{ y giving global[x] }
             topt[x]  ← link[ frompt[x] ]
             addedge[x]  ← <frompt[x],topt[x]>
             link[x]  ← comp[ topt[x] ]
             if the set is empty then link[x]  ← NULL
5.      for all x s.t. comp[x] = x
             if there is more than one y s.t. link[y]=x
             then  link[x] ← NULL
             if there is exactly one y s.t. link[y]=x
             then randomly do either (i) or (ii)
             using a different r.n. generator for each x:
             (i)   link[x]  ← NULL
             (ii)  link[y]  ← NULL
6.      for all x s.t. comp[x] = x AND link[x] ≠ NULL
             addedge[x] is to be part of the min spanning tree
             comp[x]  ← link[x]
7.      for all x
             comp[x]  ← comp[ comp[x] ]
```

Initially, in step 1, all vertices are in a component by themselves. Step 2 checks, in O(1) time, the condition for continuing to loop. Each iteration of steps 3-7 will reduce the number of components by, on the average, a quarter (to be shown). Thus only O(lg n) iterations suffice. Each iteration requires O(1) time (to be shown). Thus, total time is O(lg n).

Step 3 determines, for each x, the minimum edge connecting x to a vertex in a different component. Step 4 determines, for each component, the minimum edge connecting it to a different component. The result is a set of tree-loops (see [HCS]).

Step 5 deletes some of the edges from the tree-loops so that, among the edges that remain, no two edges are head-to-tail. Therefore, no loops can be introduced. Also, the expected number of edges that remain will be at least a quarter of those that were there originally. This can be seen from the following observation. A node deletes at most half of the edges that touch it. An edge will not be deleted only if neither of its endpoints delete it which, on the average, happens at least a quarter of the time. Therefore, the number of components will be reduced by, on the average, at least a quarter each iteration.

Step 6 adds the edges to the spanning tree and updates the component memberships of the representatives of each (previous) component. Step 7 updates the component memberships of all other vertices.

The proof of correctness of this algorithm is similar to that in [Sa].

Each of steps 3-7 can be performed in O(1) time. One non-obvious aspect to be demonstrated is that a statement of the form

```
          value  ←  min{ array[y] | expression(x,y)=TRUE }
```

can be executed in O(1) time if expression(x,y) can be evaluated in O(1) time. We show this as follows.

```
a) for all y
       best[y]  ←  FALSE
b) for all y s.t. expression(x,y)=TRUE do steps c,d
c)     best[y]  ←  TRUE
d)     for all z s.t. expression(x,z)=TRUE
                 if ( array[y]<array[z]
                      OR ( array[y]=array[z] AND y<z ) )
                 then         best[z]  ←  FALSE
e) for all y     if best[y]=TRUE   value  ←  array[y]
```

Step 5 can be executed in O(1) by the use of the following:

```
 ymin  ←  min{ y | link[y]=x } , if set is empty then NULL
 ymax  ←  max{ y | link[y]=x } , if set is empty then NULL
 if (ymin ≠ ymax) then there is more than one y s.t. link[y]=x
 else if ymin ≠ NULL then there is exactly one y s.t. link[y]=x
```

We note that the min operation requires n^2 processors. The entire procedure of forming a minimum spanning tree thus requires O(lg n) time and n^3 processors.

CONCLUDING REMARKS

We have shown that several graph problems can be solved in time O(lg n) on a WRAM. Any algorithm taking time T(n) on a WRAM can be converted into an algorithm taking time T(n)lg n on a PRAC [Ec]. Therefore, these graph problems can be solved on a PRAC in time $(\lg n)^2$.

Thus far, all of the problems that have been solved for the WRAM have time complexity O(1) or O(lg n). It would be of interest to characterize the class of problems that are solvable within each order of time complexity.

REFERENCES

[Ar] Arjomandi E. "A study of parallelism in graph theory" CS Dept. Univ. of Toronto TR 86, Dec. 1975.

[BH] Borodin A. and J.E. Hopcroft "Routing, merging and sorting on parallel models of computation" 14th SIGACT (1982), pp.337-344.

[CLC] Chin F.Y., J. Lam and I.N. Chen "Efficient parallel algorithms for some graph problems" CS Dept. Univ. of Alberta, TR 79-12, Sept. 1979, to appear Comm. ACM.

[DNS] Dekel E., D. Nassimi and S. Sahni "Parallel matrix and graph algorithms" CS Dept., Univ. of Minnesota, TR 79-10, Sept. 1979.

[Ec] Eckstein, D. "Simultaneous memory accesses", Iowa State Univ. TR 79-6, August 1979.

[FW] Fortune S. and J. Wylie "Parallelism in random access machines" 10th SIGACT (1978), pp.114-118.

[Go] Goldschlager L. "A unified approach to models of synchronous parallel machines" 10th SIGACT (1978), pp.89-94.

[HCS] Hirschberg D., A. Chandra and D. Sarwate "Computing connected components on parallel computers" Comm. ACM 22:8 (Aug. 1979), pp.461-464.

[LPV] Lev G., N. Pippenger and L.G. Valiant "A fast parallel algorithm for routing in permutation networks" IEEE Trans. on Comp., 1981.

[Pr] Preparata F.P. "New parallel sorting schemes" IEEE Trans. on Comp., July 1978.

[Sa] Savage C. "Parallel algorithms for graph theoretic problems" CSL Univ. of Illinois, Rpt. R-784, Aug. 1977.

[SV] Shiloach Y. and U. Vishkin "Finding the maximum, merging and sorting in a parallel computation model" CS Dept. Technion Israel, TR-173, March 1980.

[SV2] Stockmeyer L. and U. Vishkin "Simulation of parallel random access machines by circuits" IBM Yorktown Heights, 1982.

THE M-SHUFFLE AS AN INTERCONNECTION NETWORK FOR SIMD MACHINES*

KURT KEUTZER and EDWARD ROBERTSON
Department of Computer Science
Indiana University
Bloomington, Indiana 47405

ABSTRACT

Parallelism offers an attractive approach in the ongoing attempt to decrease the time required for computational tasks. In parallel computations, speed of communication between processors becomes preeminently important. For this reason much attention has been devoted to communication networks for parallel processors. This paper explores the use of the augmented m-shuffle as a single stage interconnection network for single instruction, multidata machines and demonstrates its applicability to a number of parallel matrix algorithms. It also characterizes algebraic properties of compositions of m-shuffles.

1. MACHINE MODEL

The model of SIMD machine used here is essentially that of [Siegel79]. It consists of N independent processing elements (PE's), each with local memory, controlled by a central control unit and interconnected by a network. The interconnection network consists of a set of interconnection functions (IF's). Each IF is a permutation of $\{0, 1, \ldots N\text{-}1\}$, the addresses of the PE's. A given element $<i,j>$ of an interconnection function **f** indicates that there is a direct connection from processor i to processor j. The execution of a data transfer instruction using **f** simultaneously transfers the contents of the output data transfer register of processor i to the input data transfer register of processor j for all $<i,j>$ in **f**.

2. THE M-SHUFFLE

The wide applicability of the perfect shuffle as an interconnection function for both single stage networks [Stone71, Wu81] and multistage networks [Lawrie75] is well known. This paper generalizes the idea of a perfect shuffle to that of an m-shuffle.

Thinking of a stack of N processors as a deck of cards, an m-shuffle is performed by cutting the deck into m piles of N/m cards and interleaving them, picking up the top card from the first stack, the top card from the second stack, ... the top card from the m-th stack, iteratively for N/m iterations. If one makes a direct connection between the i-th processor in the stack before the shuffle and the i-th processor in the stack after the shuffle then one has defined an m-shuffle interconnection function. A perfect shuffle is thus merely a 2-shuffle.

The m-shuffle IF on a network of size N, denoted ${}_N\mathrm{S}_m$ may be defined, for $m \mid N$ and $0 \leq i \leq N-1$, by

$${}_N\mathrm{S}_m(i) = \mathrm{mod}(i, N/m) \times m + \left\lfloor \frac{mi}{N} \right\rfloor$$

The m-shuffle has been used by [Patel81] as a unit for constructing the multistage delta network. The m-way shuffle, a multiple-interconnection network which is a superset of the m-shuffle, has been considered by [Valiant81]. These papers are concerned

---

* This research was supported by National Science Foundation grant MCS 80-04337

with the ability of the m-shuffle to provide random access capability either from processors to memories across a multistage network [Patel81] or from processors to processors with queues at each processor [Valiant81]. This paper does not explore the random access capabilities of the m-shuffle but instead looks at the m-shuffles ability to provide precisely the interconnections that are needed to implement many parallel matrix algorithms.

3. THE AUGMENTED M-SHUFFLE

Generally a perfect shuffle network is augmented by other communication paths. A shuffle-exchange network also has an "exchange" network which allows processors $2i$ and $2i+1$ to communicate. For m-shuffles additional connections are similarly valuable. On a network of size $N = m^n$ it is desirable to have connections available among all processors sharing the same initial $n-1$ digits. A set of processors sharing this property will be called a digit group. In the generalization of a shuffle-exchange network considered in this paper, the processor whose address is $p_{n-1} \cdots p_0$ in m-adic notation is connected to $p_{n-1} \cdots p_1 i$ for $0 \leq i \leq m-1$ by the augmenting network and to processor $p_{n-2} \cdots p_0 p_{n-1}$ by the m-shuffle. This later fact is shown in [Keutzer] by an extension of the argument from [Stone71] which shows that a perfect shuffle on a network of size 2^m transfers data from the processor whose address in binary is $i_{m-1} i_{m-2} \cdots i_0$ to processor $i_{m-2} i_{m-3} \cdots i_0 i_{m-1}$.

For large m, complete connection among digit groups of processors is too expensive in hardware, is too difficult to control and has more connections than are needed. A nice compromise allowing for fairly fast broadcasting and efficient fan-in of information is to augment the shuffle network with PM2I connections [Siegel79] for each digit group.

4. ALGORITHMS

An early application of the perfect shuffle is matrix multiplication. A straightforward application of the above-mentioned result from [Stone71] shows that if a 2^n by 2^n matrix is distributed over 2^{2n} processors in such a way that the matrix element's address in binary defines the address of the processor containing that matrix element, then the matrix can be transposed after n consecutive perfect shuffles.

An advantage of generalizing the perfect shuffle IF to an m-shuffle IF can easily be seen in matrix transposition. If the distribution of the matrix over the processors is as above but the perfect shuffle IF is replaced by a 2^n-shuffle then the transposition can be performed in constant time. As discussed in section 3, performing an m-shuffle on a network of size m^n is equivalent to performing a single cyclic left shift of the processor's address when written in m-adic. If a 2^n-shuffle IF were used on the example above then the processor whose address in 2^n-adic notation is ij would be directly connected to processor ji and the corresponding array element could be transferred by a single shuffle. Figure 1 illustrates a 4 by 4 matrix distributed over 16 processors.

The savings that a constant time matrix transposition gives in more complicated algorithms can be seen in the following, which is an algorithm for matrix multiplying two m by m matrices A and B on an augmented m-shuffle network of m^3 processors. For the sake of clarity m is assumed to be a power of two. The difficulty is that, if m is not a power of two, then the PM2I network as given in [Siegel79] is not fully defined. However, the algorithm does work correctly with an interpretation of a PM2I network in which certain connections are omitted .

Square brackets will be used in denoting processing elements; thus P[ijk] denotes the processor with m-adic address ijk and A[ijk] denotes the contents the A register of that processor. Parentheses will denote the original matrices; thus $a(i,j)$ is the element of matrix A in row i and column j. Initially $a(i,j)$ is stored in the A register of processor P[ijj] and $b(i,j)$ is stored in the B register of processor P[ijj]. The A and B registers of all other processors are zero. At the completion of the algorithm $C=A \times B$ and $c(i,j)$ is stored in the C register of processor P[ijj].

Matrix Multiply Algorithm

```
      /* PHASE I                                       Processors          */
      /* Initialize A registers                        Containing $a(i,k)$ */

                                                   /*  P[ikk]              */
  1)  m-shuffle                                    /*  P[kki]              */
  2)  m-shuffle                                    /*  P[kik]              */
  3)  PM2I broadcast     /* $\log m$ steps */      /*  P[ki*]              */
  4)  m-shuffle                                    /*  P[i*k]              */

      /* PHASE II                                      Processors          */
      /*Initialize B registers                         Containing $b(k,j)$ */

                                                   /*  P[kjj]              */
  5)  m-shuffle                                    /*  P[jjk]              */
  6)  m-shuffle                                    /*  P[jkj]              */
  7)  PM2I broadcast     /* $\log m$ steps */      /*  P[jk*]              */
  8)  m-shuffle                                    /*  P[k*j]              */
  9)  m-shuffle                                    /*  P[*jk]              */

      /*  At this point processor P[ijk] has $a(i,k)$ in A[ijk] and $b(k,j)$ in B[ijk]. */
      /*  P[ijk] is now able to do the step C[ijk] = $A(i,k) \times B(k,j)$            */
      /*  using only local memory                                                      */

      /* PHASE III                                                                     */
      /* Compute products and sum over k terms using PM2I connections                  */

  10) C[ijk] = A[ijk] $\times$ B[ijk]

  11) for l = 0 to log(m) - 1  do
  12) C[ijk] = C[ijk] + C[ij mod(k+ $2^l$,m)]

      /* At this point, C[ij*] = $\sum_{k=0}^{m-1} a[i,k] \times b[k,j]$  for $0 \le i,j \le m-1$   */
```

The above algorithm requires $\log_2 m$ unit routes for each of the broadcasts at steps 3 and 7 and another $\log_2 m$ unit routes for the summing at step 11. The total communication time for this algorithm is therefore $3\log_2 m$. This offers a constant factor improvement over the $5\log_2 m$ time of cube connected cycles (CCC) and the $10\log_2 m$ time of a shuffle exchange network [Dekel81].

A "paracomputer" algorithm for transitive closure was given in [Hirschberg76]. This algorithm uses the fact that the transitive closure of an m by m bit matrix A can be computed by iteratively squaring $(A+I)$ for $\lceil \log_2 m \rceil$ iterations. An "ultracomputer" version of Hirschberg's transitive closure algorithm can be given using the matrix multiplication algorithm given above. The resulting algorithm maintains the $\log^2 m$ time bound of the original algorithm. (Paracomputers are idealized shared-memory machines; ultracomputers require memory access through interconnection networks [Schwartz81].)

[Reghbati78] gives a number of paracomputer graph algorithms, including connected components and strongly connected components, which are reducible to iterations of transitive closure. The same m-shuffle implementation may be applied to these problems while retaining the $\log^2 m$ paracomputer time bounds. Implementation of other graph algorithms having matrix multiply at their base, such as those given in [Dekel81], should be straightforward.

Earlier we saw that an augmented m-shuffle implementation of the matrix multiply algorithm offered a constant factor speed up over either CCC or shuffle exchange. The augmented m-shuffle overs other advantages as well. One advantage of the augmented m-shuffle implementation is that it requires neither the number of processors nor the matrix size to be a power of two. To multiply a 65 by 65 matrix in log time on a shuffle exchange or CCC network would require either padding the matrix out to an 128 by 128 matrix requiring 128^3 processors as opposed to 65^3 on the augmented m-shuffle or "folding" the matrix which would degrade the time bounds.

Another advantage of implementing matrix algorithms on an m-shuffle is that they often require no masking. For a discussion of masking in SIMD machines see [Siegel77]. The problem of finding the transitive closure of a 2^k by 2^k bit matrix illustrates how masking greatly increases the complexity of each processing element. To obtain the $O(\log_2 m)$ time bounds this requires 2^{3k} processors each with local memory. However the local memory need only store a few bits of data and the processor need only be able to take the AND of two bits of information. In such a configuration the additional overhead of an address register of $3k$ bits and the logic to interpret that information for masking eclipses the requirements for performing the computation.

5. THEORETICAL RESULTS

This section of this paper gives a number of theoretical properties of m-shuffles. Their proofs are found in [Keutzer].

Theorem 1: If $N = k \times l \times m$ for positive integers k, l, m, N, then

$$ {}_N\mathrm{S}_{l\times m} = {}_N\mathrm{S}_l \circ {}_N\mathrm{S}_m. $$

Corollary 2: If $m^z \mid N$ then ${}_N\mathrm{S}_{m^z} = {}_N\mathrm{S}_m^z$.

Definition: If $m \mid N$ then ${}_N\mathrm{S}_m^0 = \mathbf{I}$ (the identity shuffle) and ${}_N\mathrm{S}_m^k = {}_N\mathrm{S}_m^{k-1} \circ {}_N\mathrm{S}_m$.

Definition: If there exists a positive integer l such that ${}_N\mathrm{S}_z^l = {}_N\mathrm{S}_y$, then we say that ${}_N\mathrm{S}_z$ *emulates* ${}_N\mathrm{S}_y$ and denote that by ${}_N\mathrm{S}_z \geq_e {}_N\mathrm{S}_y$. If ${}_N\mathrm{S}_z \geq_e {}_N\mathrm{S}_y$ and ${}_N\mathrm{S}_y \geq_e {}_N\mathrm{S}_z$ we denote this relationship by ${}_N\mathrm{S}_z \equiv_e {}_N\mathrm{S}_y$.

For example, if $N = 2^3 \times 3^5$ then ${}_N\mathrm{S}_6 = {}_N\mathrm{S}_2 \circ {}_N\mathrm{S}_3$ This implies that a perfect shuffle on N can emulate a 4- or 8-shuffle and can in composition with a 3-shuffle emulate all well defined shuffles on N.

Theorem 3: If $N = m^z$ then ${}_N\mathrm{S}_m^y = {}_N\mathrm{S}_m^{\mathrm{mod}(y,z)}$.

Theorem 4: Let $N=m^z$ and let y, q, r and x be positive integers such that $y=qx+r$ and $0 \leq r < x$, then ${}_N S_{m^y}$ can emulate ${}_N S_{m^x}$ iff

a) there exists an l such that $x \equiv_z y \times l$ and

b) $\gcd(y,z) | x$.

Both theorem 3 and theorem 4 give results regarding networks in which the size of the network and the shuffles themselves are powers of m. Theorem 3 tells us that y consecutive left shifts are equivalent to $\mathrm{mod}(y,z)$ consecutive left shifts because if $y=qz+r$ and $0 \leq r < m$ each of the qz left shifts bring us back to our original position and only the $r = \mathrm{mod}(y,z)$ left shifts change the position of the data.

Theorem 4 gives the requirements for emulation of one m-shuffle connection scheme by another on networks satisfying the hypothesis. Theorem 4 tells us for instance that if $N = 2^{17}$ then ${}_N S_{2^9} \geq_e {}_N S_2$ but if $N=2^{2k}$ then a 4-shuffle cannot emulate a perfect shuffle on that network.

At this point it is valuable to introduce some algebraic formalism. Recall that ${}_N S_m$ is a permutation on $\{0, 1, \ldots N\text{-}1\}$. We may now state:

Theorem 5: ${}_N S^* = \{ {}_N S_m : m | N \}$ is a finite abelian group over composition of mappings.

Theorem 6: ${}_N S^*_m = \{ {}_N S^k_m : m | N$ and k is a positive integer$\}$ is also a finite abelian group over composition of mappings.

Theorem 7: Let $N = l \times m$ and let k be the order of ${}_N S^*_m$, then ${}_N S_l \equiv_e {}_N S_m$ and each emulation takes at most k steps.

Figure 3 illustrates the sets ${}_{12} S^*$, ${}_{12} S^*_6$ and ${}_{12} S^*_2$ and shows the results of theorem 7 when applied to $N = 2 \times 6$, namely that ${}_{12} S_2 \equiv_e {}_{12} S_6$.

While most effective when tailored to a specific application, the above results show that m-shuffles have a great degree of vitality. As mentioned in section 2, [Patel81] has used m-shuffles as interstage links in delta networks. For instance an $a^n \times b^n$ delta network is constructed by using n a-shuffles as interstage links. The theoretical results of this section show that there are a number of alternative shuffle configurations which may be used as interstage links. This allows for considerably more flexibility in network design.

6. CONCLUSION

This research may be seen in a more general light as attempting to define the primitive operations of paracomputer algorithms and to determine what time and processor losses are incurred, if any, in their translation into ultracomputer algorithms. This paper shows that on an augmented m-shuffle network the primitive operation on m by m matrices which computes

$a(j,i) = a(i,j)$ for $0 \leq i,j \leq m-1$

can be performed in constant time with m^2 processors and that

$$c(i,j) = \sum_{k=0}^{m-1} a(i,k) \times b(k,j) \text{ for } 0 \leq i,j \leq m-1$$

can be performed in $3\log_2 m$ time on an augmented m-shuffle network of m^3 processors.

The ongoing development of new paracomputer algorithms guarantees an abundance of areas for further research. In particular the area of lower bound results for implementing basic paracomputer operations on ultracomputers seems entirely neglected.

REFERENCES

[Dekel81] E. Dekel, D. Nassimi, S. Sahni, "Parallel Matrix and Graph Algorithms," *SIAM J. Comput., 10,* 4, 657-675 (1981).

[Hirschberg76] D. Hirschberg, "Parallel Algorithms for the Transitive Closure and the Connected Component Problems," *Proc. 8th Annual Symp. on Theory of Comp.,* May 1976, 55-57.

[Keutzer] K. Keutzer, *Ph.D Thesis,* Department of Computer Science, Indiana University (in prepation).

[Lawrie75] D. Lawrie, "Access and Alignment of Data in an Array Processor," *IEEE Trans. on Comput.,* C-24, 12, 1145-1155 (1975).

[Patel81] J. Patel, "Performance of Processor-Memory Interconnections for Multiprocessors," *IEEE Trans. on Comput.,* C-30, 10, 771-780 (1981).

[Reghbati78] E. Reghbati, D. Corneil, "Parallel Computations in Graph Theory," *SIAM J. Comput., 7,* 2, 230-237 (1978).

[Schwartz80] J. T. Schwartz, "Ultracomputers," *ACM Trans. on Prog. Lang. and Systems, 2,* 4, 484-521 (1980).

[Siegel77] H. J. Siegel, "Analysis Techniques for SIMD Machine Interconnection Networks and the Effect of Processor Address Masks," *IEEE Trans. on Comput.,* C-26, 2, 153-161 (1977).

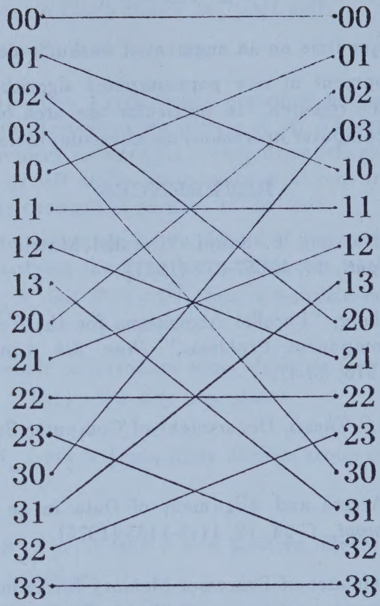
[Siegel79] H. J. Siegel, "A Model of SIMD Machines and Comparison of Various Interconnection Networks," *IEEE Trans. on Comput.,* C-28, 12, 907-917 (1979).

[Stone71] H. S. Stone, "Parallel Processing with the Perfect Shuffle," *IEEE Trans. on Comput.,* C-20, 2, 153-161 (1971).

[Valiant81] L. Valiant. "Universal Schemes for Parallel Communication," *Proc. of the 13th Annual ACM Symposium on Theory of Computing,* May 1981, 263-267.

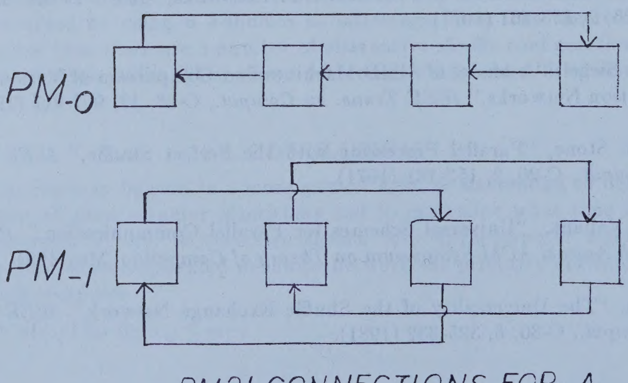
[Wu81] C. Wu, "The Universality of the Shuffle Exchange Network," *IEEE Trans. on Comput.,* C-30, 5, 325-332 (1981).

FIG. 1


00
01
02
03
10
11
12
13
20
21
22
23
30
31
32
33
00
01
02
03
10
11
12
13
20
21
22
23
30
31
32
33

$_{16}S_4$

FIG. 2



PM2I CONNECTIONS FOR A DIGIT GROUP OF SIZE FOUR

$_{12}S_6^0$	$_{12}S_6^1$	$_{12}S_6^2$	$_{12}S_6^3$	$_{12}S_6^4$	$_{12}S_6^5$	$_{12}S_6^6$	$_{12}S_6^7$	$_{12}S_6^8$	$_{12}S_6^9$	$_{12}S_6^{10}$
0	0	0	0	0	0	0	0	0	0	0
1	2	4	8	5	10	9	7	3	6	1
2	4	8	5	10	9	7	3	6	1	2
3	6	1	2	4	8	5	10	9	7	3
4	8	5	10	9	7	3	6	1	2	4
5	10	9	7	3	6	1	2	4	8	5
6	1	2	4	8	5	10	9	7	3	6
7	3	6	1	2	4	8	5	10	9	7
8	5	10	9	7	3	6	1	2	4	8
9	7	3	6	1	2	4	8	5	10	9
10	9	7	3	6	1	2	4	8	5	10
11	11	11	11	11	11	11	11	11	11	11
$_{12}S_2^{10}$	$_{12}S_2^9$	$_{12}S_2^8$	$_{12}S_2^7$	$_{12}S_2^6$	$_{12}S_2^5$	$_{12}S_2^4$	$_{12}S_2^3$	$_{12}S_2^2$	$_{12}S_2^1$	$_{12}S_2^0$
$_{12}S_1^1$	$_{12}S_6^1$	$_{12}S_3^1$						$_{12}S_4^1$	$_{12}S_2^1$	$_{12}S_{12}^1$

Figure 3.

PARSING DYCK LANGUAGES ON PARALLEL MACHINES

R.M. MATTHEYSES and C.M. FIDUCCIA
General Electric R+D Center
Schenectady, New York 12345

ABSTRACT

An algorithm is presented to decide the membership question for a string x with respect to a Dyck set D_k over k types of parentheses in $O(\log|x|)$ time using $|x|$ processors. If x is a member of D_k, its parse tree is also constructed, in additional time that is independent of the input, by using the data structure constructed during the recognition phase. These bounds are obtained on a parallel PRAM computer in which processors have access to a common memory. Simultaneous access to a memory location is allowed for reading but simultaneous writes or writes and reads are not allowed.

INTRODUCTION

A Dyck language D_k over an alphabet A_k of k types of parentheses consists of the set of all properly nested, balanced parentheses strings in which left and right parentheses of a balanced pair are of the same type [2]. D_k is defined as the context-free language generated by the productions

$$\begin{array}{ll} S \rightarrow (_i\ )_i & 1 \leq i \leq k \\ S \rightarrow (_i S\ )_i & 1 \leq i \leq k \\ S \rightarrow S\ S & \end{array}$$

where $(_i$ is the left parenthesis of type i and $)_i$ is the corresponding right parenthesis. These two parentheses are said to be of the same type and said to match.

We present a parallel algorithm for parsing strings in D_k which is designed for the PRAM model of computation [1]. A PRAM computer consists of a shared random access memory and a collection of synchronous processing elements. The processing elements each have an index which is set to an initial value. The index may be referenced but not changed by the processing elements. In addition, each processor has a small number of registers for use in performing calculations. The processors use and communicate through the shared random access memory in a very liberal fashion. During one memory cycle each processor can read or write any cell in the shared memory. Simultaneous access to a cell by several processors is permitted. However, if one processor is writing into a memory cell an attempt by any other processor to write or read that cell is considered a conflict. Conflicts are forbidden. The input to the PRAM appears in the first n memory locations.

The PRAM model was studied by Fortune and Wyllie [1]. They showed that any algorithm which runs on a sequential machine using space S leads to an algorithm which runs on a PRAM in time S. Since there is an obvious sequential algorithm for recognizing D_k in log space, there is a corresponding PRAM algorithm which runs in log time; however, the algorithm provided by their construction requires a number of processing elements that is exponential in S. We present a recognition algorithm for Dyck languages that is efficient in both time and processor requirements. Moreover, our algorithm generates the parse tree for valid Dyck strings in additional time which is independent of the length of the input string.

In the next section we discuss the central idea which led to the Dyck language algorithm. In subsequent sections we present the algorithm in detail. Throughout the remainder of the paper we will use the input string

$$(([ ]) [ ])$$

as an example in discussions of the algorithm.

REDUCED FORMS FOR D_1

Before tackling general Dyck languages, we consider the recognition problem for D_1 to explain the central idea that leads to the more general algorithm.

Given an arbitrary string x, of left and right parentheses in A_1^*, we can determine whether they are properly nested and balanced by repeated application of the single rewrite rule

$$() \rightarrow \lambda$$

where λ represents the empty string. A string x is in reduced form when the rewrite rule is no longer applicable. It is easy to see that reduced forms consist of i unmatched right parentheses followed by j unmatched left parentheses $)^i(^j$. No other forms are stable. We represent this form by the pair of natural numbers [i,j]. The simplicity of the replacement rule makes it easy to establish the fact that every string has a unique reduced form.

To decide whether x is properly nested and balanced, we need only test whether its reduced form is $[0,0] = \lambda$. We have thus transformed the recognition problem into one of computing reduced forms in parallel. To do this we introduce the binary infix operator $\downarrow : N^2 \times N^2 \rightarrow N^2$ that combines two reduced forms to yield the reduced form of their concatenation.

$$)^i(^j \downarrow)^I(^J =)^m(^n$$

$$m = i + \max(I-j,0)$$
$$n = J + \max(j-I,0)$$

We assume that this operation can be performed in constant time by a processing element of the PRAM. This is not unreasonable, since all values are never larger than the length |x| of the input string.

As a consequence of the uniqueness of reduced forms, the ↓ operator is associative - a fact that can be proved directly from the above definition. This means that $(N^2, \downarrow, [0,0])$ is a monoid and that the reduction map, call it ρ, is a monoid homomorphism from A_1^* into this monoid.

The associativity of ↓ gives us complete freedom in reducing the input string. Recognition is now nothing more than the computation of ρ(x) in the new monoid. Noting that a single parenthesis maps to one of the reduced forms [0,1] or [1,0], we need only start with the sequence of these forms, in N^2, corresponding to x and construct any binary tree on it to compute the desired value ρ(x) at the root. By choosing a tree of height

$O(\log|x|)$, the reduction can be accomplished in $O(\log|x|)$ time using $O(|x|)$ processing elements. Note that the computation does not require the power of the PRAM and could be done on a tree machine. On the other hand it is easy to show [3] that $O(|x|/\log|x|)$ processing elements are sufficient on the PRAM, while maintaining the $O(\log|x|)$ time behavior.

THE ALGORITHM

The main algorithm of this paper recognizes and parses Dyck strings. This is accomplished in three phases. The first two phases deal with the recognition problem. The third phase generates the parse tree, provided the input is a Dyck string.

The first two phases are based on the easily established fact that a string x is in D_k iff x is in D_1, when its parentheses are treated simply as left or right parentheses without regard for type, and each balancing pair of x is of the same type. A balancing pair of parentheses is a pair generated by the one of the type-subscripted production given in the Introduction.

Phase one checks for membership in D_1 by constructing a minimum-depth binary tree with the original input string x at its leaves. Leaves that are left parentheses are labeled [0,1]; those that are right parentheses are labeled [1,0]. Each internal node of the tree is labeled with the reduced form of the leaves of its induced subtree, as determined by the labels of its sons. If the root is labeled $[0,0] = \lambda$, the algorithm proceeds to phase two; otherwise, it rejects the input string and halts.

During phase two, processors are assigned to the individual elements of the input string. Using the labeled tree constructed by phase one, each processor finds a balanced pair of parentheses. Once each pair is determined, they are checked to see if they match; i.e., if they are of the same type. The results of the individual matchings are combined, again in log $(|x|)$ time, to determine whether all pairs match. If so, the input string is a Dyck string and the algorithm proceeds to phase three; otherwise, the string is rejected and the algorithm halts.

During phase three, a processing element is assigned to each left parenthesis of the original input string. Each processor decides, by examining a fixed neighborhood around its balancing right parenthesis, which of a small number of parse tree fragments it must generate. The root of the fragment will be in a location that corresponds to the left parenthesis represented by the processor. The parse tree generation is done in constant time, independently of its size or shape. No use is made of the minimum-depth reduction tree constructed by phase one. The three phases are presented in the following sections.

BUILD THE TREE OF REDUCED FORMS

During the first phase of the algorithm, parentheses are interpreted as either left or right without regard to their types. The "typeless" string is processed, to decide if it is in D_1, using the $\downarrow$ operator. During this phase a minimum-depth binary tree is constructed whose leaves correspond to the input string. The tree is constructed in a linear array using implicit addressing for the father/son relations. For the sake of exposition we pad the original input x with enough λ's to make its length a power of two. The first phase of the algorithm starts with processors at the leaves. They construct the tree and label the nodes in the tree with the reduced form of the parentheses string at the leaves of their corresponding subtrees. The entire string is well-formed, in D_1, iff the root receives the label [0,0].

```
Algorithm BUILD TREE:
Input: A parentheses string of length N in S[1:N].
       N' the smallest power of 2 not less than N.
Output: A tree, Red[1:2N'-1], with reduced forms labeling
         the nodes. For a node i, Father(i)=⌊i/2⌋
         Left(i)=2i and Right(i)=2i+1, i is a leaf if i ≥ N'.
Comment: The following code is executed in parallel by all
          processors i where 1≤i≤N'.

   1. Leaf ← i + N' - 1
      Node ← Leaf
      If i > N Then
         Red[Node] ← [0,0]
      Elseif S[i] is left paren Then
         Red[Node] ← [0,1]
      Else
         Red[Node] ← [1,0]
      Endifs

   2. For Log(N') iterations Do
         If Node is a left son Then
            Node      ← Father(Node)
            Red[Node] ← Red[Left(Node)] ↓ Red[Right(Node)]
         Else
            Skip
         Endif
      Endfor
```

For the example, after completion of phase one of the algorithm, the memory will contain the tree in Figure 1. We proceed to phase two only if the string is determined to be well-formed by phase one.

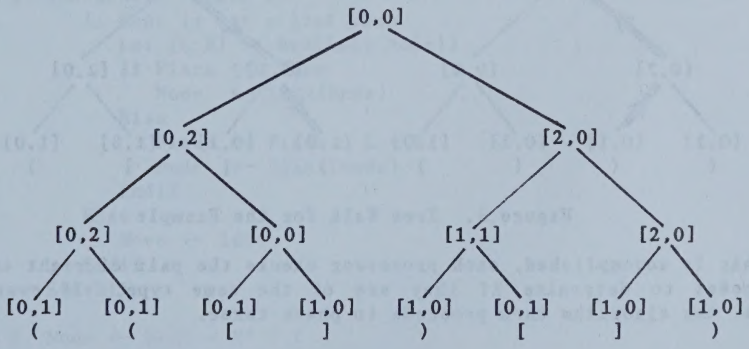


Figure 1. Tree of Reduced Forms for the Example

WALKING THE TREE OF REDUCED FORMS

During phase two, it is determined if balancing pairs of parentheses, of the well-formed input x, are of the same type. This is accomplished by assigning a processor to each left parenthesis and letting it walk up the tree of reduced forms, and then back down, toward the balancing right parenthesis. The walk is guided by the values deposited at the nodes of the tree by phase one. On the way up, the processor keeps track of the minimum number of right parentheses required to cancel its own left parenthesis and all the other left parentheses to its right. Once the processor detects the cancellation of its own parenthesis, it reverses direction and starts to walk down the tree, keeping track of the position of the right parenthesis that caused the cancellation. This is the balancing right parenthesis it was seeking.

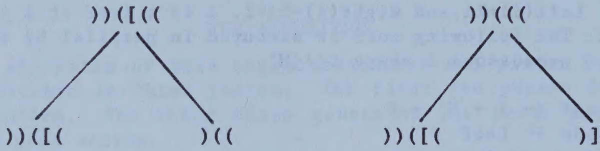


Figure 2. Typical Steps in a Tree Walk

In each case in Figure 2, the square bracket denotes the place being tracked by a single processor. In the case on the left, the processor continues up the tree. In the case on the right, having found the right parenthesis that balances its left parenthesis, the processor starts down the right subtree of its father. The rest of the walk is merely to determine at what leaf it resides. The leaf that it finds will correspond to the balancing right parenthesis in the input string. The walk of one processor performed during phase two of the algorithm, when applied to the example, is illustrated in Figure 3. Simultaneously, paths will be traced corresponding to each left parenthesis.

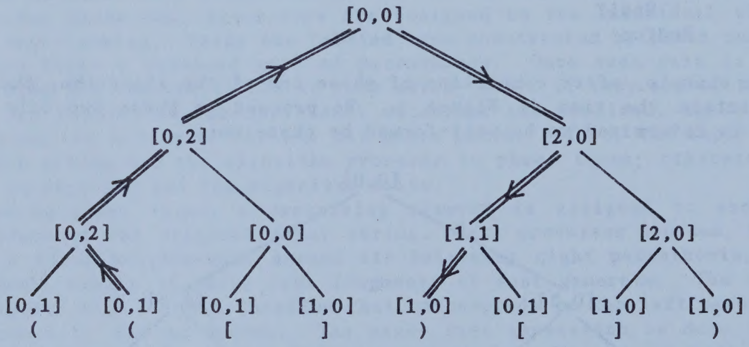


Figure 3. Tree Walk for the Example

Once this is accomplished, each processor checks the pair of right and left parentheses to determine if they are of the same type. If every pair matches, the algorithm then proceeds to phase three.

Algorithm TREE WALK:

Input: The original input string S[1:N], N, and N'. The value of Leaf and The tree Red[1:2N'-1] produced by TREE BUILD.

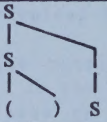
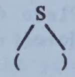
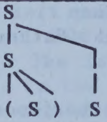
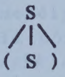
Output: The array Bal[1:N]. Bal[I] is the index, in x, of the parenthesis that balances S[I]. The answer as to whether the input string was Dyck, in location 1 of the array Dyck[1:N].

Comments: This code is performed by all processors i, $1 \le i \le N$.

```
1. If Red[Leaf] = [0,1] Then Do steps 2-5
                             Else Skip

2. Node  ← Leaf
   Place ← 1
   Move  ← up

3. While Move = up Do
      If Node is a left son Then
         Let [L,R] be Red[Right(Father(Node))]
         If Place $\le$ L Then
            Move  ← down
            Node  ← Right(Father(Node))
         Else
            Place ← Place - L + R
            Node  ← Father(Node)
         Endif
      Else
         Node ← Father(Node)
      Endif
   Endwhile

4. While Move = down Do
      If Node is not a leaf Then
         Let [L,R] be Red[Left(Node)]
         If Place $\le$ L Then
            Node  ← Left(Node)
         Else
            Place ← Place - L + R
            Node  ← Right(Node)
         Endif
      Else
         Move ← idle
      Endif
   Endwhile

5. Node ← Node - N' + 1
   Leaf ← Leaf - N' + 1
   If S[Leaf] and S[Node] are of same type Then
      Dyck[Leaf] ← True
      Dyck[Node] ← True
      Bal[Leaf]  ← Node
      Bal[Node]  ← Leaf
   Else
      Dyck[Leaf] ← False
      Dyck[Node] ← False
   Endif

6. Reduce Dyck[1:N] using "AND"
```

BUILDING THE PARSE TREE

The third phase - parse tree construction - is somewhat more complicated. It relies upon the knowledge, gained during phase two, concerning which parentheses form balancing pairs. The processor corresponding to a balancing pair of parentheses constructs the appropriate portion of the parse tree. All the possible cases that must be considered by each processor are given in the following table.

Parse Tree Alternatives	First character to right of pair is left parenthesis	
	yes	no
Balancing pair is adjacent in string		
Balancing pair not adjacent in string		

The non-terminal symbols at the leaves of the tree fragments correspond to roots of subtrees generated by other matching pairs of parentheses. All that is required is that the root of each fragment be in an easy to determine location, such as in a memory location with an index that corresponds to the location in the input string of the left parenthesis that it generates. Figure 4 presents the case when the balancing pair is not adjacent in the input string and there is a left parenthesis to the right of it. A represents the substring between the balancing pair which is a Dyck string in its own right. B is the longest substring immediately to the right of the balancing pair that is a Dyck string. It is the substring whose first character is the left parenthesis to the right of the balancing pair.

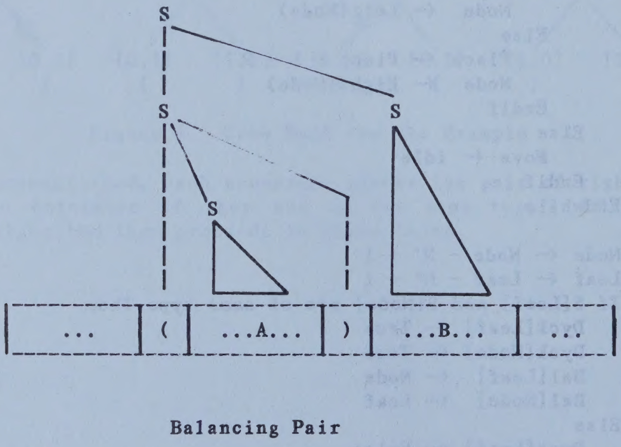


Figure 4. Parse Tree Construction

```
Algorithm PARSE TREE CONSTRUCT:
Input: The array Bal[1:N] of balancing parentheses positions
       and the original string S[1:N].
Output: The array P[1:N] containing the parse tree.  Each entry
        consists of four fields: P[I,Label]=label of node I,
        P[I,Left]=left son pointer, P[I,Right]=right son pointer, and
        P[I,Mid]=middle son pointer.  P[1] is the root of the tree.
Comment: The prefix operator # returns a pointer to the object to
         which it is applied.  The following code is executed by
         all processors i with 1≤i≤N such that S[i] is a left
         parenthesis.

   1. If Bal[i]<N and S[Bal[i]+1] is a left parenthesis Then
         P[i,Label]        ← "S"
         P[i,Left]         ← #P[Bal[i]]
         P[i,Right]        ← #P[i+1]
         P[i,Mid]          ← Null
         P[Bal[i],Label]   ← "S"
         P[Bal[i],Left]    ← #S[i]
         P[Bal[i],Right]   ← #S[Bal[i]]
         P[Bal[i],Mid]     ← If i+1=Bal[i] Then Null
                                               Else #P[i+1] Endif
      Else
         P[i,Label]      ← "S"
         P[i,Left]       ← #S[i]
         P[i,Right]      ← #S[Bal[i]]
         P[i,Mid]        ← If i+1=Bal[i] Then Null
                                             Else #P[i+1] Endif
      Endif
```

Phase three of the algorithm will produce the parse tree in Figure 5 when applied to the example input string.

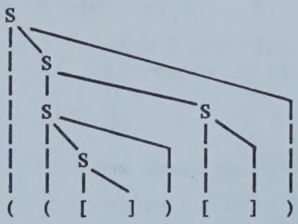


Figure 5. Parse Tree for the Example

The non-terminal symbols above each left parenthesis and the edges to their descendants correspond to the production constructed by the processor associated with that parenthesis. Upon completion of phase three, the root of the parse tree will be in P[1].

REFERENCES

1. Fortune, S., and J. Wyllie, "Parallelism in Random Access Machines", Proc. Tenth Annual ACM Symposium on the Theory of Computing, 1978, pp. 114-118.

2. Hopcroft, J.E., and J.D. Ullman, Introduction to Automata Theory, Languages, and Computation, Addison Wesley, Reading, Massachusetts, 1979, p. 142.

3. Kuck, D.J., The Structure of Computers and Computations, Volume 1, John Wiley and Sons, New York, 1978, p. 91.

QUANTIZATION WITH NON-DIFFERENCE BASED DISTORTION MEASURES

EFREN F. ABAYA
Department of Electrical Engineering
University of Texas at Austin
Austin, Texas 78712

and

GARY L. WISE
Departments of Electrical Engineering and Mathematics
University of Texas at Austin
Austin, Texas 78712

ABSTRACT

Suppose that a sequence of probability distributions $\{F_n\}$ converges weakly to a distribution function F. Does the sequence of optimal quantizers for the F_n's converge to an optimal quantizer for F? If so, do the respective distortions converge to the optimal distortion for F? Sufficient conditions are given to guarantee the convergence for vector quantizers with non-difference based distortion measures. These results are used to demonstrate the strong consistency of a certain class of estimators for optimal quantizers, which are generated by quantizing empirical distribution functions. As a side result the existence of optimal quantizers is also shown.

1. INTRODUCTION

An N-level k-dimensional vector quantizer is a mapping $Q: \mathbb{R}^k \to \mathbb{R}^k$ which assigns the input vector x to an output vector Q(x) chosen from a finite set of N distinct vectors $\{y_i : y_i \in \mathbb{R}^k,\ i=1,2,\dots,N\}$. Generally, the quantizer input is modelled as a random vector X described by a k-variate distribution F. A measure of quantizer performance is the distortion function

$$D(Q,F) = \int d(x,Q(x))\, dF(x). \qquad (1)$$

Traditional cost functions d(x,y) were based on the quantization error; for example, the r-th power cost function $\|x-y\|^r$, for which several design algorithms are available [1-3]. It is known that these cost functions are not always satisfactory, especially when held up against subjective evaluations. Recently, more complicated cost functions have been investigated in an effort to obtain more meaningful distortion measures. Sharma [4,5] used a continuous cost function $f(x, \|x-y\|)$ in a problem of encoding video signals . The matrix-based cost function $(x-y)^T R(x) (x-y)$, where R(x) is a positive definite matrix, is used in the linear prediction of speech [6]. An algorithm applicable to this and all of the previous cost functions is proposed and analyzed in [7-8]. Reference [8] contains a good description of distortion measures used in quantization.

The problem in quantizer optimization is to minimize (1) for a specified dimension k and number of output vectors N. If the levels y_i are known, then the best quantizer is constructed by the "nearest neighbor assignment rule" [8]: Q(x) is the output vector y_i which minimizes $d(x,y_i)$. The manner in which ties are broken is unimportant. This rule will be adopted throughout the rest of this paper. Then the optimization problem can be stated as: Find a set $\{y_i, i=1,2, \ldots, N\}$ which minimizes the distortion (1). If such a set exists, the resulting quantizer is said to be optimal for F.

The distribution F is rarely known in practice. One might then take independent samples from F and design an optimal quantizer for the empirical distribution of the observations. One then hopes that this design closely approximates the optimal quantizer for F. Linde, Buzo, and Gray [8] developed a design algorithm based on this idea. In previous papers [9,10] the authors studied the convergence of empirically derived quantizers, using difference distortion measures. In this paper we generalize those results to non-difference based distortions.

Specifically, we consider the following problem. Suppose that a sequence of distribution functions $\{F_n\}$ converges weakly to a distribution function F. Let Q_n be an optimal N-level quantizer for F_n. We pose the following questions:

(Q1) Do the distortions $D(Q_n,F_n)$ converge to $D(Q,F)$, where Q is an optimal quantizer for F?

(Q2) Does $\{Q_n\}$ converge weakly to some quantizer Q? If so, is Q optimal for F?

Sufficient conditions for these statements to be true are described in Sect. 4, and applied to empirical distributions in Sect. 5. The cost function considered in this paper is described in Sect. 2. The concept of convergence of quantizer sequences is discussed in Sect. 3.

2. THE COST FUNCTION

The cost of quantizing a vector x to a level y is a non-negative function $d(x, y)$ on $\mathbb{R}^k \times \mathbb{R}^k$ satisfying the following conditions, hereafter referred to as *Assumption A:* (1) $d(x, y) \geq 0$ with equality when $x=y$; (2) for every y, $d(x, y)$ is continuous in x; (3) for every x, $d(x, y)$ is a convex function of y; and (4) for each bounded x-set S_x, $\lim d(x, y) = \infty$ as $\| y \| \to \infty$, uniformly over S_x.

Assumptions (1) - (3) are quite natural, and are usually required to obtain convergence of numerical algorithms for optimal quantization. Property 4 is needed in the development to obtain uniform convergence of $d(x, y)$ as y goes to an infinite limit, and entails that a finite ball $\{x: \| x \| \leq M\}$ is uniformly poorly approximated by large vectors y, i.e., $\| y \| \gg M$. Also, the following properties will be used in the development: (2') for every x, $d(x, y)$ is a continuous function of y; and (5) for each bounded x-set S_x and bounded y-set S_y, $|d(x, y) - d(x, y_0)| = O(\| y-y_0 \|)$ as $y \to y_0$, whenever $x \in S_x$ and $y, y_0 \in S_y$. These additional properties follow from assumptions (2) and (3) by properties of convex functions [11].

From assumption (2), it follows that $d(x, Q(x))$ is a continuous function of x. Assumption (2') implies that for a fixed x, $d(x, Q(x))$ is a continuous function of the output levels $\{y_1, y_2, \ldots, y_N\}$.

In the development, we will need an additional restriction:
Assumption B: For each y, $d(x, y)$ is uniformly integrable with respect

to $\{F_n\}$, i.e.,

$$\lim_{a \to \infty} \sup_{n} \int_{\|x\|>a} d(x, y) \; dF_n(x) = 0.$$

Assumptions A and B are understood to hold throughout the rest of the paper.

3. CONVERGENCE OF QUANTIZERS

We say that a sequence of N-level quantizers Q_n converges weakly to the quantizer Q if $Q_n(x) \to Q(x)$ at all continuity points of Q. In general we will allow the limit quantizer Q to have $M \leq N$ levels. Recall that under the nearest neighbor assignment rule, a quantizer Q is completely represented by the set of its output vectors $\{y_1, y_2, \ldots, y_M\}$. Then the output levels of the Q_n, under a suitable reindexing, converge to the respective output levels of Q. For example, under the mild restriction that each y_i is a continuity point of Q, it follows that $Q_n(y_i) \to Q(y_i)$ so that $\{Q_n(y_1), Q_n(y_2), \ldots, Q_n(y_M)\}$ comprise a suitable ordering of M of the levels of Q_n. The other N-M levels become inconsequential as far as the limiting behavior of $Q_n(x)$ is concerned.

An equivalent definition of quantizer convergence may be given in terms of the output levels. Let $B(y;\varepsilon)$ be a closed ball in $\mathbb{R}^k$ of radius ε centered on y. Denote the origin of $\mathbb{R}^k$ by $\tilde{0}$. Let

$$S(\varepsilon) = \bigcup_{i=1}^{M} B(y_i;\varepsilon) \quad \cup B^c(\tilde{0};1/\varepsilon) \tag{2}$$

where the superscript denotes complementation. Then Q_n converges weakly to Q if and only if for every $\varepsilon > 0$, there exists an N_0 such that whenever $n > N_0$, all of the output levels of Q_n lie in $S(\varepsilon)$, and moreover, each $B(y_i;\varepsilon)$ contains at least one output level of Q_n. We will represent weak convergence of quantizers by $Q_n \to Q$.

If $\|Q_n(x)\| \to \infty$ for every x, we say that the sequence Q_n diverges. Equivalently, for $r > 0$, there exists an N_0 such that whenever $n > N_0$, all of the output levels of Q_n lie outside $B(\tilde{0};r)$.

Suppose $Q_n \to Q$ and Q_n has output levels $y_1^n, y_2^n, \ldots, y_N^n$. Then it follows that $d(x, Q_n(x)) \to d(x, Q(x))$ for all x. This convergence is even uniform over bounded x-sets, as we now show.

Lemma 1. If $Q_n(x) \to Q(x)$ weakly, then $d(x, Q_n(x)) \to d(x, Q(x))$ uniformly over bounded x-sets.

Proof. Denote the output levels of Q by $y_1, y_2, \ldots, y_M$ where $1 \leq M \leq N$. Let S_x be a bounded x-set. By taking ε small enough in (2), it is possible to have $Q_n(x) \in \bigcup_{i=1}^{M} B(y_1:\varepsilon)$ for $n > N_0$ and $x \in S_x$. This follows from assumption (A4). Then by property (5), for $n > N_0$

$$|d(x, Q_n(x)) - d(x, Q(x))| = O(\|Q_n(x) - Q(x)\|). \tag{3}$$

Now if x is a continuity point of Q, the right hand side is $O(\varepsilon)$. If x is a boundary point of the partitions, the argument gets more involved, but we still get the same conclusion. For example, if x is a boundary point of the partitions for y_1 and y_2, it must be true that $d(x, y_1) = d(x, y_2)$. Then $Q_n(x)$ can be separated into subsequences converging to y_1 and y_2 respectively. Applying (3) to each subsequence gives the result. Thus $d(x, Q_n(x))$ converges uniformly over S_x. QED

4. DEVELOPMENT

In this section we examine the questions (Q1) and (Q2) put forth in Section 1 and establish four propositions which give sufficient conditions for an affirmative answer. First we establish a useful lemma.

Lemma 2. Assume that $F_n \to F$ weakly, and $Q_n(x) \to Q(x)$ weakly. Assumption B implies that $d(x, Q_n(x))$ is uniformly integrable with respect to $\{F_n\}$, i.e.,

$$\lim_{a \to \infty} \sup_{n} \int_{\|x\| > a} d(x, Q_n(x)) \; dF_n(x) = 0. \tag{4}$$

Proof. Denote the output levels of Q_n by $y_1^n, y_2^n, \ldots, y_N^n$ and the output levels of Q by $y_1, y_2, \ldots, y_M$, $1 \leq M \leq N$. As discussed in the previous section, these may be indexed so that $y_1^n \to y_1$. Since by the nearest neighbor assignment rule $d(x, Q_n(x)) \leq d(x, y_1^n)$ for all n, it is sufficient to show that $d(x, y_1^n)$ is uniformly integrable with respect to $\{F_n\}$, i.e.,

$$\lim_{a \to \infty} \sup_{n} \int_{\|x\| > a} d(x, y_1^n) \; dF_n(x) = 0. \tag{5}$$

By assumption $y_1^n \to y_1$. Hence there exists some closed ball, $B(y_1;r)$ of radius r containing all the y_1^n's. This ball is itself contained in the interior of a hypercube $R(y_1)$. We may limit our consideration in (4) to those a's large enough that $R(y_1)$ is contained within $B(\tilde{0};a)$. Let y be a point inside $B(y_1;r)$ and x outside $B(\tilde{O};a)$. The line joining x and y pierces $R(y_1)$ at a point z, such that y is between x and z. Recall that by Assumption A, $d(x,x) = 0$ and $d(x,y)$ is convex in y. It follows that

$$d(x,y) \leq d(x,z).$$

Denote the vertices of $R(y_1)$ by $z_1, z_2, \ldots, z_{2k}$. Again by convexity we have

$$d(x,z) \leq \sum_{i=1}^{2^k} d(x,z_i).$$

Since $y_1^n \in B(y_1;r)$ for every n, the last two inequalities imply that

$$d(x,y_1^n) \leq \sum_{i=1}^{2^k} d(x, z_i) \qquad n = 1, 2, \ldots .$$

Invoking Assumption B gives (5), so the lemma is proved. QED

The above lemma implies that the quantizers Q_n all have finite distortions. The first two propositions below are the principal results about convergence of distortions. Notice that no optimality conditions are imposed upon the Q_n's or Q. This simply affirms the intuitive notion that the distortion (1) is a "smooth" function of Q and F.

Proposition 1. Suppose that $F_n \to F$ weakly and that a sequence of N-level quantizers $\{Q_n\}$ converges weakly to a quantizer Q. If Assumptions A and B are satisfied, then $D(Q_n, F_n) \to D(Q,F)$.

Proof. Recall from Sect. 2 that $d(x, Q(x))$ is continuous. By Lemma 1 and a theorem in Billingsley [12, p.34], it follows that $G_n \to G$ weakly where G_n and G are the distributions of $d(X, Q_n(X))$ and $d(X,Q(X))$ when X has the distributions F_n and F, respectively. A standard integral convergence theorem [12, p.32], justified by Lemma 2, then gives

$$\lim_{n \to \infty} \int d(x, Q_n(x))\, dF_n(x) = \int d(x, Q(x))\, dF(x) < \infty,$$

which is the conclusion of the proposition. It takes a little bit of manipulation to recast our definition (4) of uniform integrability into the conventional one, e.g. [12, p.32], but it is possible since $d(x,Q(x))$ is continuous and the convergence of $d(x,Q_n(x))$ is uniform. We omit the details QED

Proposition 2. Suppose that $F_n \to F$ weakly and that a sequence of N-level quantizers $\{Q_n\}$ diverges. If Assumptions A and B are satisfied, then $D(Q_n, F_n) \to \infty$.

Proof. Let $B(\tilde{O};r)$ be a closed ball whose F-probability exceeds one-half, and whose boundary has F-probability zero. Such a ball always exists.

The F_n probability of $B(\tilde{O};r)$ converges to its F-probability [12, p.23] so that for sufficiently large n, the respective F_n probabilities are at least one half.

Since $\{Q_n\}$ diverges, all of the y_i^n's diverge to infinity. By Assumption (A4), there exists N_0 so that for $x \in B(\tilde{O};r)$ we have $d(x, Q_n(x)) > M$ whenever $n > N_0$, where M is an arbitrarily large number. Thus, for n sufficiently large

$$\int d(x, Q_n(x))\, dF_n(x) \geq M/2.$$

Since M is arbitrary, it follows that $D(Q_n, F_n) \to \infty$. QED

The above two propositions may be used to investigate the behavior of sequences of (globally) optimal quantizers. The results are stated in two propositions. Using Props. 1 and 2 above and an obvious modification of the proofs in [10], we arrive at the following:

Proposition 3. Under the conditions of Prop. 1, if Q_n is an optimal N-level quantizer for F_n, then Q is optimal for F.

Proposition 4. Under the conditions of Prop. 3, every subsequence of $\{Q_n\}$ contains a further subsequence which converges (weakly) to some quantizer.

The previous results are summarized in the following theorem, which serves to answer the two questions posed in Sect. 1. The proof follows straightforwardly from the above propositions.

Theorem 1. Suppose that $F_n \to F$ weakly and that Q_n is an optimal N-level quantizer for F_n. If Assumptions A and B are satisfied, then every subsequence of $\{Q_n\}$ contains a further subsequence which converges weakly to an optimal quantizer for F. Moreover, $D(Q_n, F_n)$ converges to the optimal distortion for quantizing F with N output levels.

5. APPLICATIONS

As an application of the previous theorem, we consider the design of a quantizer for an unknown k-variate distribution F. Independent, identically distributed samples $X_1, X_2, \ldots, X_n$ are drawn from the distribution F. A plausible procedure is to estimate F from the data, and from this design a quantizer using any of the algorithms mentioned in Section 1. One particularly simple estimator of F is the empirical distribution $\hat{F}_n$ [12, p.103]. Let us call an optimal quantizer for an empirical distribution based on samples $X_1, X_2, \ldots, X_n$ a sample quantizer and denote it Q_n. The question we wish to consider is the following. Suppose that the sample size n increases to infinity. The Glivenko-Cantelli Theorem [13] asserts that $\hat{F}_n$ converges uniformly to F (therefore weakly, also) with probability 1. Does the sequence Q_n of sample quantizers converge to an optimal quantizer Q for F?

For example, if $d(x, y) = (x-y)^2$ and N=1 output level, the Q_n is simply the sample mean $\sum X_i/n$ which converges with probability 1 to the mean of F. In the general case, Q_n may be considered to be an estimator of the optimal quantizer Q. The question posed above then asks about the strong consistency (with probability 1 convergence to the optimal Q) of the estimator. This property is greatly to be desired of any procedure for constructing optimal quantizers.

This problem is a simple application of Theorem 1. As noted above, $\hat{F}_n \to F$ weakly. The only hypothesis which needs verification is Assumption B, concerning uniform integrability of d(x, y) with respect to the empirical distributions $\hat{F}_n$. Since d(x, y) is non-negative, Assumption B holds (with probability 1) if, and only if, [12, p.32]

$$\lim_{n \to \infty} \int d(x, y)\ d\hat{F}_n(x) = \int d(x, y)\ dF(x) \quad \text{w.p.1}, \tag{6}$$

or if, and only if,

$$\lim_{n \to \infty} \frac{1}{n} \sum_{i=1}^{n} d(X_i, y) = \int d(x, y)\ dF(x) \quad \text{w.p.1.} \tag{7}$$

Eq. (7) follows from the Strong Law of Large Numbers [14, p.239] whenever the right hand side is finite. Unfortunately, we cannot conclude from Thm. 1 that $Q_n \to Q$ weakly unless the optimal quantizer for F is unique. However, the sample distortions $D(Q_n, \hat{F}_n)$, which may be regarded as estimates of D(Q,F), are strongly consistent. We summarize these observations as a theorem.

Theorem 2. Let Q_n be an N-level sample quantizer for a set of independent observations $X_1, X_2, \ldots, X_n$ drawn from the distribution F. Suppose Assumption A is satisfied, and in addition $\int d(x, y)\ dF(x)$ is finite for every y. Then, with probability 1, every subsequence of $\{Q_n\}$ contains a further subsequence which converges weakly to an optimal quantizer for F. If the optimal quantizer for F is unique, then Q_n is a strongly consistent estimator. In either case, $D(Q_n, \hat{F}_n)$ converges with probability 1 to the optimal distortion for quantizing F with N output levels.

The strong consistency of Q_n, when F possesses a unique optimal quantizer, was shown by Pollard [15] for difference-based distortion measures. This result extends [15] in two ways. First, the cost function is more general. Second, the case that the optimal quantizer for F is not unique is handled in a natural manner.

In all of the previous discussions, we have tacitly assumed the existence of all of the optimal quantizers involved. We can now prove that this is correct.

Theorem 3. Suppose Assumption A is satisfied, and in addition $\int d(x, y)\ dF(x)$ is finite for every y. Then an optimal N-level quantizer exists for every positive integer N.

Proof. By Thm. 2, it is sufficient to show that an (optimal) sample quantizer exists for every empirical distribution $\hat{F}_n$. For then, the limit Q of any convergent subsequence of sample quantizers Q_{n_k} is optimal for F. Such a sequence exists with probability 1.

It is easier to prove a stronger statement: An optimal N-level quantizer exists if F has compact support. Since any empirical distribution has compact support, this suffices to prove the theorem.

Let $B(\tilde{0};r)$ be a closed ball in $\mathbb{R}^k$ with F-probability 1. By Assumption (A2), there exists a positive number M such that $d(x, \tilde{0}) \leq M$ for $x \in B(\tilde{0};r)$. By Assumption (A4), there exists a positive number R such that $d(x, y) > M$ for $x \in B(\tilde{0};r)$ and $||y|| \geq R$. Then for purposes of optimization, it is sufficient to restrict attention to levels for which $||y|| \leq R$. For if a quantizer had a level y_1 with $||y_1|| > R$, we can reduce the distortion by replacing y_1 by the origin $\tilde{0}$. This holds true for all the N levels of Q.

Now D(Q, F) is a continuous function of the output levels $y_1, y_2, \ldots, y_N$ (Prop. 1). In the compact ball $B(\tilde{0};R)$, it therefore achieves a minimum, and this gives an optimal quantizer for F. This completes the proof of the theorem. QED

ACKNOWLEDGEMENT

This research was supported by the Air Force Office of Scientific Research under Grant AFOSR-81-0047.

REFERENCES

[1] J.D. Bruce, "On the Optimum Quantization of Stationary Signals," *IEEE International Convention Record*, Pt. I, pp. 118-124, 1964.

[2] S.P. Lloyd, "Least Squares Quantization in PCM," Bell Telephone Labs Memorandum, Murray Hill, N.J., 1957; reprinted in *IEEE Trans. Inform. Theory*, IT-28 (1982), pp. 129-137.

[3] J. Max, "Quantizing for Minimum Distortion," *IRE Trans. Inform. Theory*, IT-6 (1960), pp. 7-12.

[4] D.K. Sharma, "Design of Absolutely Optimal Quantizers for a Wide Class of Distortion Measures," *IEEE Trans. Inform. Theory*, IT-24 (1978), pp. 693-702.

[5] D.K. Sharma and A.N. Netravali, "Design of Quantizers for DPCM Coding of Picture Signals," *IEEE Trans. Comm.*, COM-25 (1977), pp. 1267-1274.

[6] F. Itakura, "Minimum Prediction Residual Principle Applied to Speech Recognition," *IEEE Trans. Acoust., Speech, and Signal Processing*, ASSP-23 (1975), pp. 67-72.

[7] R.M. Gray, J.C. Kieffer and Y. Linde, "Locally Optimal Block Quantizer Design," *Inform. Cont.*, 45 (1980), pp. 178-198.

[8] Y. Linde, A. Buzo and R.M. Gray, "An Algorithm for Vector Quantizer Design," *IEEE Trans. Comm.*, COM-28 (1980), pp. 84-95.

[9] E.F. Abaya and G.L. Wise, "Convergence of Vector Quantizers with Application to the Design of Optimal Quantizers," *Proc. 19th Annual Allerton Conf. on Comm., Control, and Computing*, Monticello, IL, Sept. 30, Oct. 1-2, 1981, pp. 79-88.

[10] E.F. Abaya and G.L. Wise, "Convergence of Vector Quantizers with Applications to Optimal Quantization," to appear in *SIAM J. Applied Math.*

[11] R.T. Rockafellar, *Convex Analysis*, Princeton: Princeton University Press, 1970, Sect. 10.

[12] P. Billingsley, *Convergence of Probability Measures*, John Wiley and Sons, Inc., New York, 1968.

[13] L.P. Devroye, "A Uniform Bound for the Deviation of Empirical Distribution Functions," *J. Multivariate Anal.*, 7 (1977), pp. 594-597.

[14] M. Loeve, *Probability Theory*, D. Van Nostrand Company, Inc., Princeton, 1963.

[15] D. Pollard, "Quantization and the Method of K-means," *IEEE Trans. Inform. Theory*, IT-28 (1982), pp. 199-205.

OPTIMUM QUANTIZER PERFORMANCE FOR A CLASS OF NON-GAUSSIAN MEMORYLESS SOURCES[†]

N. FARVARDIN and J. W. MODESTINO
Electrical, Computer and Systems Engineering Department
Rensselaer Polytechnic Institute
Troy, New York 12181

ABSTRACT

In this paper we study the performance of optimum quantizers subject to an entropy constraint for a wide class of memoryless sources. For a general distortion criterion, we develop necessary conditions for optimality and describe a recursive algorithm for obtaining the optimum quantizer. Under a mean-square error criterion, the performance of entropy encoded uniform quantization of memoryless Gaussian sources is well-known to be within 0.255 bits/sample of the rate-distortion bound at relatively high rates. Here we demonstrate that similar performance can be expected for a wide range of carefully parameterized memoryless sources. Indeed, for the cases considered, the worst case performance is observed to be less than 0.3 bits/sample from the rate-distortion bound. Furthermore, it is shown that the performance of optimum quantizers generally improves at low bit-rates.

I. INTRODUCTION

The simplest and most common source coding technique is zero-memory quantization of the real-valued outputs of an appropriately defined source. This technique has been and continues to be a popular method of digitizing analog signals. The wide-spread use of quantizers has naturally spurred a significant activity in optimizing their performance [1]-[6].

The entropy H of the output of a quantizer is the minimum amount of information necessary to be transmitted in order to be able to reconstruct the quantizer output with arbitrarily small error. An optimum quantizer with entropy constraint is defined to have a minimum distortion $D_Q(H)$ between the input real-valued sequence and the corresponding discrete-valued outputs, for a given output entropy H. The $D_Q(H)$ curves provide bounds on the performance achievable by one-dimensional quantization. In addition, the merits of quantization as opposed to other means of source coding can be ascertained by comparing $D_Q(H)$ to Shannon's rate-distortion function $R(D)$, or equivalently the distortion-rate function $D(R)$[7].

Goblick and Holsinger [8] have shown that, in the case of minimum mean-square digitization of memoryless Gaussian sources, quantizers with uniformly spaced levels have entropies that exceed the rate-distortion function lower bound by approximately 0.25 bits/sample at relatively high rates. It is also known from the work of Gish and Pierce [9] that the optimum quantizers with an entropy constraint should tend toward uniformity in the limit of small distortion (large entropy) and large number of levels. More recently, Noll and Zelinski [10] have investigated the quantization of memoryless sources with uniform, Gaussian, Laplacian, and Gamma distributions. The performance of the optimum quantizer, subject to an entropy constraint, is compared to the rate-distortion function of the corresponding source in the low bit-rate region. It is concluded that the optimally attainable distortion for small entropy values is significantly higher than the lower bound previously established by Gish and Pierce [9] for small distortions.

In this paper we have investigated the performance of optimum quantizers for a wide class of carefully parameterized memoryless non-Gaussian sources. We have demonstrated that uniform threshold quantizers perform

---

[†] This work was supported in part by ONR under Contract No. N00014-75-C-0281.

effectively as well as do optimum quantizers. In addition, we have shown, in contrast to the previously reported results [10], that for most cases the performance of the optimum quantizer actually improves at low bit-rates. In fact, we have found the worst case performance to be only 0.3 bits/sample from the rate-distortion function lower bound and this has been the case for a source with uniform distribution which, in addition, has exhibited other anomalies to be described. We have used the uniform distribution case to demonstrate the possibility of multiple optimum, as well as asymmetric quantizers. Similar to Goblick and Holsinger [8], we have developed explicit asymptotic relations for the rate-distortion performance of optimum quantizers in terms of parameters describing the source density.

II. NOTATION AND PRELIMINARY ANALYSIS

We investigate the one-dimensional or zero-memory quantization of a discrete-time zero-mean memoryless stationary process $\{X_t\}$ with probability density function (p.d.f.) $p(x)$, and variance $\sigma^2=E\{X^2\}$. An N-level quantizer is a function $q_N(\cdot)$ which maps the source output $x\epsilon R$ into one of N values $Q_1,Q_2,\ldots,Q_N$ each in R. This operation can be described in terms of threshold levels $T_1,T_2,\ldots,T_{N-1}$, partitioning the support of p into N disjoint and exhaustive regions, such that $x\epsilon R$ is mapped to $y=q_N(x)\epsilon\{Q_1,Q_2,\ldots,Q_N\}$ according to

$$q_N(x) = \sum_{\ell=1}^{N} Q_\ell I_\ell(x) \quad , \tag{1}$$

where

$$I_\ell(x) = \begin{cases} 1 \quad , & x\epsilon(T_{\ell-1},T_\ell] \\ 0 \quad , & \text{elsewhere} \quad ; \quad \ell=1,2,\ldots,N \end{cases} \tag{2}$$

is the indicator function of the set $(T_{\ell-1},T_\ell]$ and $(T_0,T_N]=(a,b]$ is the support of p.

Here, the T_ℓ are called the quantization thresholds and the Q_ℓ are called the reconstruction (representation) levels.

An N-level quantizer converts each source output independently of other outputs and yields an average distortion

$$D_N(\underline{T};\underline{Q})= \sum_{\ell=1}^{N} \int_a^b d(x,Q_\ell)I_\ell(x)p(x)dx \quad , \tag{3}$$

where $d(\cdot,\cdot)$ is a nonnegative distortion measure, and $\underline{T}$ and $\underline{Q}$ are given by $\underline{T}=[T_1,T_2,\ldots,T_{N-1}]^T$ and $\underline{Q}=[Q_1,Q_2,\ldots,Q_N]^T$, respectively.

Upon defining

$$P_\ell \overset{\Delta}{=} \Pr\{T_{\ell-1}<X \leq T_\ell\} = \int_a^b p(x)I_\ell(x)dx; \; \ell=1,2,\ldots,N, \tag{4}$$

the probability that the source output assumes a value in the ℓ'th partition, the entropy associated with the quantizer output will be given by

$$H_N(\underline{T}) = - \sum_{\ell=1}^{N} P_\ell \log_2 P_\ell, \text{ bits/sample} \quad . \tag{5}$$

Definition 1: An N-level quantizer q_N is said to be optimum (Lloyd-Max) if it minimizes the average distortion for a fixed number of levels N.

Definition 2: An N-level quantizer q_N is said to be optimum subject to an entropy constraint if it minimizes the average distortion while the output entropy is held below a given value.

If in the above definition, the constraint on the number of quantization levels is removed, the resulting quantizer will provide a lower bound to the rate-distortion performance of all zero-memory quantization schemes.

Throughout this work, for succinctness, we refer to the quantizer of Definition 1 as the "Lloyd-Max quantizer", while that of Definition 2 will be referred to as the "optimum quantizer".

III. NECESSARY CONDITIONS FOR OPTIMALITY

In what follows we develop necessary conditions on the quantization thresholds T_ℓ, $\ell=1,2,\dots,N-1$, and reconstruction levels Q_ℓ, $\ell=1,2,\dots,N$, so as to yield an optimum N-level quantizer with entropy constraint $H \le H_0$. We denote by $D_N(H_0)$, the minimum average distortion incurred by N-level optimum quantizer with $H \le H_0$. More formally,

$$D_N(H_0) = \inf D_N(\underline{T};\underline{Q}) \quad , \tag{6}$$

where the infimum is taken over all $\underline{T}\varepsilon R^{N-1}$, $\underline{Q}\varepsilon R^{N}$ such that $H_N(\underline{T}) \le H_0$.

In light of Definition 2, the problem of optimum quantizer design with entropy constraint can be formulated as the following nonlinear programming problem:

$$\text{Minimize:} \quad D_N(\underline{T};\underline{Q}) = \sum_{\ell=1}^{N} \int_{T_{\ell-1}}^{T_\ell} d(x,Q_\ell)p(x)dx \quad , \tag{7a}$$

$$\text{Subject to:} \quad H_N(\underline{T}) = -\sum_{\ell=1}^{N} P_\ell \log_2 P_\ell \le H_0 \quad . \tag{7b}$$

Necessary conditions for the solution to this problem follow from the Kuhn-Tucker Theorem [11]. Specifically, letting $(\underline{T}^*;\underline{Q}^*)$ be a local (relative) solution of (7), there must exist a scalar $\lambda \ge 0$ such that[†]

$$\nabla D_N(\underline{T}^*;\underline{Q}^*) + \lambda \nabla H_N(\underline{T}^*) = \underline{0} \quad , \tag{8a}$$

and

$$\lambda[H_N(\underline{T}^*) - H_0] = 0 \tag{8b}$$

Condition (8a) yields 2N-1 nonlinear simultaneous equations to be solved. The observant reader will note that H_N is only a function of $\underline{T}$ and hence the choice of $\underline{Q}$ has no effect on the value of output entropy. Therefore, one can generally find the optimum output levels for any given set of thresholds. Specifically, the optimum quantization levels must satisfy

$$\frac{\partial}{\partial Q_\ell} D_N(\underline{T};\underline{Q})\Big|_{(\underline{T}^*;\underline{Q}^*)} = \int_{T^*_{\ell-1}}^{T^*_\ell} \left[\frac{\partial}{\partial Q_\ell} d(x,Q^*_\ell)\right] p(x)dx = 0; \ \ell=1,2,\dots,N. \tag{9}$$

Equation (9) is the generic form of the well-known result

$$Q^*_\ell = \frac{1}{P^*_\ell} \int_{T^*_{\ell-1}}^{T^*_\ell} x p(x)dx \quad , \tag{10}$$

obtained by Max [2] for the mean-square distortion measure, with

$$P^*_\ell = \int_{T^*_{\ell-1}}^{T^*_\ell} p(x)dx \quad . \tag{11}$$

After appropriate simplifications, (8) can be written in the following way:

$$\lambda \ln(P^*_{\ell+1}/P^*_\ell) = d(T^*_\ell, Q^*_{\ell+1}) - d(T^*_\ell, Q^*_\ell) \ ; \ \ell=1,2,\dots,N-1, \tag{12a}$$

and

$$\lambda[H_N(\underline{T}^*) - H_0] = 0 \quad , \tag{12b}$$

[†] The operator $\nabla(\cdot)$ indicates the gradient of a scalar function with respect to the indicated arguments.

where the Q^*_ℓ satisfy (9).

Equation (12a) is the generalized version of the same result for the r'th power absolute-error distortion measure derived by Berger [14]. In the special case, when $d(x,y)=(x-y)^2$, (12a) reduces to

$$\lambda \ell n(P^*_{\ell+1}/P^*_\ell)=(Q^*_{\ell+1}-Q^*_\ell)(Q^*_{\ell+1}+Q^*_\ell-2T^*_\ell); \quad \ell=1,2,\ldots,N-1 \quad , \qquad (13)$$

previously noted by Berger [4], and Noll and Zelinski [10], where now the Q^*_ℓ are given by (10).

In general, to find $\{T^*_\ell\}$ and $\{Q^*_\ell\}$ that satisfy the necessary conditions (12) we must solve a set of N-1 nonlinear simultaneous equations given by (12a) for the $\{T^*_\ell\}$ while simultaneously satisfying (12b). Note that the corresponding $\{Q^*_\ell\}$ can be obtained from the resulting $\{T^*_\ell\}$ through (9). Solving this problem is an arduous task even for the simple case of squared-error distortion measure, as noted by Berger [4], [14], since both P^*_ℓ and Q^*_ℓ are rather complicated functions of T^*_ℓ and $T^*_{\ell-1}$.

Recall that for an N-level Lloyd-Max quantizer, the distortion is minimized without regard to the output entropy of the quantizer. Let the entropy of an N-level Lloyd-Max quantizer be denoted by $H_{N,LM}$. This suggests that for $H_0 \geq H_{N,LM}$, the optimum N-level quantizer with entropy constraint H_0, coincides with the Lloyd-Max quantizer. But this implies that $\lambda=0$ in (8), or equivalently (12). On the other hand, if an optimum quantizer exists with $H_0<H_{N,LM}$ then it cannot coincide with the Lloyd-Max quantizer and hence λ cannot vanish for $H_0<H_{N,LM}$. Thus for $H_0<H_{N,LM}$ the necessary condition (12b) yields $H_N(\underline{T}^*)=H_0$ and the optimum occurs at the boundary of the constraint set. Furthermore, it is relatively easy to establish that the distortion of an optimum quantizer is a nonincreasing function of the output entropy. These observations will be used in what follows.

IV. ALGORITHMIC APPROACH

We now proceed and develop a means of methodically obtaining optimal quantizers by describing a recursive method for solving (12). In what follows, for the mean-squared error, we suggest an algorithm which is used in the present study to find the optimum quantizer.

Algorithm (Generalized Max Algorithm):

In this method, we solve the set of nonlinear simultaneous equations (12) directly. We shall, temporarily, assume that in (12) λ is known and solve the system of equations (12) in the following steps:

(0) Given the number of levels N, and fixed λ, set $T_0=a$, the greatest lower bound (G.L.B.) of the support of $p(\cdot)$, choose an initial value for T_1 and set $\ell=1$.

(1) For the present values of $T_{\ell-1}$ and T_ℓ, use (12a) to find $T_{\ell+1}$. If $\ell \leq N-1$, replace ℓ by $\ell+1$ and go to (1). Otherwise continue.

(2) If T_N obtained in step (1) is equal to b, the least upper bound (L.U.B.) of the support of $p(\cdot)$, the initial guess for T_1 is good and the resulting $\underline{T}$ and $\underline{Q}$ satisfy the necessary conditions for optimality. Otherwise, T_1 is a poor guess. Change T_1, set $\ell=1$, and go to (1).

In the first place, we recognize that setting $\lambda=0$ in (13) immediately reveals the similarity of this algorithm with the Max algorithm. As noted previously, solving (13) for a fixed ℓ is difficult, and in fact for any given $T_{\ell-1}$ and T_ℓ, there might be up to two solutions for $T_{\ell+1}$ in (13).

This observation, which is based on empirical evidence, has also been reported by Berger [4], [14]. The roots of (13) can be obtained using a root-seeking method for solving nonlinear equations. Thus, for any initial guess of T_1, there could be two solutions for T_2, four solutions for T_3, and finally 2^{N-1} solutions for T_N. One could envision this situation as a tree whose nodes at a given depth are labeled by the solutions for the corresponding threshold level (i.e., the root node is labeled by T_1, the first order nodes are labeled by the two solutions of T_2 and so on). The pecularity of this method is that we do not generally know how T_N depends upon T_1, even if we know that the solution for T_N is unique, and hence the strategy of choosing T_1 in successive attempts is indeterminate. Moreover, we have made the interesting observation that for different values of T_1, different nodes of the (N-1)'st order can yield the optimal threshold vector. This finding, contrary to the claim that the smaller solution at each level results in the optimum solution [4], manifests the complexity associated with this algorithm due to an exhaustive search which must be made along all paths of the tree. Recall also, that in the algorithm λ is fixed, and hence to find the absolute optimum quantizer with entropy constraint H_0, we must search among all quantizers obtained from the algorithm with different values of λ having entropy H_0, and select the one with the smallest average distortion.

We have used the proposed algorithm to find optimum quantizers for a class of memoryless sources to assess the relative performance penalties associated with zero-memory quantization of real-world non-Gaussian sources described in the subsequent section.

V. MEMORYLESS NON-GAUSSIAN SOURCE MODELS

In what follows we assume that the p.d.f. of the assumed memoryless source is chosen from one of the following classes of parametric distributions (cf. [12] for additional details).

A.) *Generalized Gaussian Distribution:*

This class of parameterized distribution is described according to

$$p(x) = \left[\frac{\alpha\eta(\alpha,\beta)}{2\Gamma(1/\alpha)} \right] \exp\{-[\eta(\alpha,\beta)|x|]^{\alpha}\} , \tag{14}$$

where $\eta(\alpha,\beta) \triangleq \beta^{-1}[\Gamma(3/\alpha)/\Gamma(1/\alpha)]^{\frac{1}{2}}$ with $\alpha>0$ describing the exponential rate of decay, β a positive quantity representing a scale parameter, and $\Gamma(\cdot)$ is the gamma function. The variance of the associated random variable is given by $\sigma^2=\beta^2$. For $\alpha=2$ we have the Gaussian distribution while for $\alpha=1$ we have the Laplacian distribution.

The generalized Gaussian distribution with values of α in the range $0.1<\alpha<1.0$ have been found to provide a useful model of broad-tailed processess as, for example, atmospherical impulsive noise phenomenon (cf. [15]). Typical behavior of p.d.f.'s as a function of normalized input are illustrated in Fig. 1 for selected parameter choices.

In what follows we will have occasion to make use of the differential source entropy h_s defined by

$$h_s = - \int_{-\infty}^{\infty} p(x)\log_2 p(x)dx \quad . \tag{15}$$

For the generalized Gaussian distribution, h_s can be conveniently expressed in terms of the source parameters as

$$h_s=-\log_2[\frac{\alpha\eta(\alpha,\beta)}{2\Gamma(1/\alpha)}] + 1/\alpha\ell n2; \text{ bits/sample.} \tag{16}$$

B.) *Generalized Cauchy Distribution:*

This class of parameterized distributions are described according to

$$p(x)=\left[\frac{\alpha\eta(\alpha,\beta)}{2\Gamma(1/\alpha)}\right]\left[\frac{\nu^{-1/\alpha}\Gamma(\nu+1/\alpha)}{\Gamma(\nu)}\right]\left\{1+[\eta(\alpha,\beta)|x|]^{\alpha}/\nu\right\}^{-(\nu+1/\alpha)}, \qquad (17)$$

β represents a scale parameter, and α,ν are positive parameters completely defining this distribution. The generalized Cauchy distribution exhibits algebraic decay of the tails vis-à-vis the exponential decay associated with the generalized Gaussian distribution. For $\alpha=2$ and $\nu=1/2$ we have the Cauchy distribution.

The variance associated with the generalized Cauchy distribution is given by $\sigma^2=\beta^2[\nu^{2/\alpha}\Gamma(\nu-2/\alpha)/\Gamma(\nu)]$ which exists only for $\alpha\nu>2$. Also, the differential entropy of this source is given by

$$h_s=-\log_2\left\{\left[\frac{\alpha\eta(\alpha,\beta)}{2\Gamma(1/\alpha)}\right]\left[\frac{\nu^{-1/\alpha}\Gamma(\nu+1/\alpha)}{\Gamma(\nu)}\right]\right\}+(\nu+1/\alpha)[\Psi(\nu+1/\alpha)-\Psi(\nu)]/\ell n2;\ \text{bits/sample}, \qquad (18)$$

where $\Psi(\cdot)$ is the Euler's psi function defined by $\Psi(x)\overset{\Delta}{=}d\ell n\Gamma(x)/dx$.

The generalized Cauchy distribution can provide a useful model for a variety of non-Gaussian broad-tailed stochastic phenomena.

C.) *Generalized Beta Distribution:*

Unlike the generalized Gaussian and Cauchy distributions defined for all real x, the generalized Beta distribution is confined to a finite interval. In particular, the probability density function in this case is given by

$$p(x)=\left[\frac{\alpha\eta(\alpha,\beta)}{2\Gamma(1/\alpha)}\right]\left[\frac{\nu^{-1/\alpha}\Gamma(\nu+1+2/\alpha)}{\Gamma(\nu+1+1/\alpha)}\right]\cdot\left\{1-[\eta(\alpha,\beta)|x|]^{\alpha}/\nu\right\}^{(\nu+1/\alpha)}, \qquad (19)$$

for $|x|\leq\nu^{1/\alpha}/\eta(\alpha,\beta)$ where α,ν are positive parameters which, together with the scale parameter β, define this distribution. The variance associated with the generalized Beta distribution is given by $\sigma^2=\beta^2[\nu^{2/\alpha}\Gamma(\nu+1+2/\alpha)/\Gamma(\nu+1+4/\alpha)]$, which is finite for all positive α and ν. The differential entropy in this case is given by

$$h_s=-\log_2\left\{\left[\frac{\alpha\eta(\alpha,\beta)}{2\Gamma(1/\alpha)}\right]\left[\frac{\nu^{-1/\alpha}\Gamma(\nu+1+2/\alpha)}{\Gamma(\nu+1+1/\alpha)}\right]\right\}+(\nu+1/\alpha)[\Psi(\nu+1+2/\alpha)-\Psi(\nu+1+1/\alpha)]/\ell n2;\ \text{bits/sample}. \qquad (20)$$

VI. NUMERICAL RESULTS

We have used two different quantization schemes to digitize the output of a memoryless source with p.d.f. chosen from the class of generalized distributions of the preceding section. Mean-square error is chosen to be the distortion criterion throughout.

A.) *Optimum Quantizer with Entropy Constraint:*

For each p.d.f. chosen from the class of generalized distributions, we have employed the algorithm described in Section IV to obtain the optimum N-level quantizer. Here, $D_N(H_0)$ will denote the smallest distortion attainable by an N-level quantizer with fixed output entropy H_0, and $D_Q(H_0)\overset{\Delta}{=}\inf_N D_N(H_0)$.

B.) *Uniform Threshold Quantizer:*

As a second strategy for quantizer design, we have investigated the behavior of a symmetric uniform threshold quantizer. That is, the inner thresholds are chosen such that $T_{\ell+1}-T_\ell=\delta\sigma$, $\ell=1,2,\ldots,N-2$, where δ is the step size normalized to the standard deviation of the input process.

The motivation for considering uniform quantizers has been the observation made by Wood [15] regarding the optimality of uniform quantizers for Gaussian sources. Similar observations were made later by Berger [4], and Noll and Zelinski [10]. In our scheme, although the thresholds are taken to be uniformly spaced, we will take the reconstruction levels according to

(10) so as to minimize the average distortion for a given stepsize. We denote by $D_N^*(H_0)$ the smallest distortion associated with an N-level uniform threshold quantizer with output entropy H_Q, while $D_Q^*(H_0) \overset{\Delta}{=} \inf_N D_N^*(H_0)$ specifies the lower bound on the performance of uniform threshold quantizers.

Notice that the class of generalized distributions consists of symmetric densities. As noted by Berger [14], and proved in [13], if N(number of levels) is even and the p.d.f. is symmetric, no symmetric quantizer can yield an output entropy less than 1 bit/sample. Since we have confined ourselves to symmetric uniform quantizers, we have chosen N odd to be able to obtain entropies below 1 bit/sample.

The above two quantization strategies have been applied to selected cases of generalized distributions for N=3,5,9,17, and 33. Also, the rate-distortion performance of an asymmetric two-level quantizer is studied for the same source densities, by varying its single threshold level.

Before we present and discuss the results, let us consider the case of a uniform distribution as it possesses some interesting properties.

In this case, as shown in [13] there is more than one quantizer yielding the same distortion and entropy, and hence optimal. This establishes the fact that the optimum quantizer is not unique. Furthermore, unless $H_0 \geq H_{N,LM}$, the optimum quantizer turns out to be nonuniform, which is contrary to the common belief that optimum quantizers are nearly uniform for symmetric distributions.

The rate-distortion performance $D_N(H_0)$ of optimum N-level quantizers, and $D_N^*(H_0)$ for uniform threshold quantizers, together with the rate-distortion function R(D) of a uniformly distributed source are illustrated in Fig.'s 2 and 3, respectively. In these figures we have also presented the Gish-Pierce asymptotic result. Specifically, Gish and Pierce [9] have shown that under weak assumptions on the density function and the distortion measure, the optimum quantizer with a large number of levels tends toward uniformity at high rates. Moreover, they have established a relationship between the rate and distortion of optimum quantizers for small distortions given by $\hat{D}(H_0) = 1/12\ 2^{2(h_s - H_0)}$, where h_s is the differential source entropy defined by (15) and the number of levels is assumed to be large.

By using the expressions for h_s given by equations (16), (18) and (20) for the three different classes of densities to evaluate $\hat{D}(H_0)$, one could find an asymptotic formula for the rate distortion performance of optimum quantizers at high rates.

Careful comparison of these results reveals that there are examples where for a given rate and fixed N the optimum quantizer performs slightly better than the uniform quantizer. For instance, for N=5, $D_5^{-1}(0.0625)=$ 2.0 bits/sample, while $D_5^{*-1}(0.0625)=2.197$ bits/sample. However, comparing Fig.'s 2 and 3 reveals that although for a fixed N the optimum quantizer might outperform the uniform threshold quantizer for the same value of N, when the constraint on the number of levels is removed the optimum and uniform quantizers will become equivalent, i.e., $D_Q(H_0)=D_Q^*(H_0)$.

We now concentrate on the class of generalized distributions and the resulting rate-distortion performance. The performance of optimum quantizers together with uniform threshold quantizers are illustrated in Fig.'s 4-7 for selected parameters of the generalized Gaussian distribution illustrated in Fig. 1. Similar results are obtained for the two other generalized distributions. For further results one may refer to [13]. For all of these cases the corresponding rate-distortion function R(D) has been calculated and sketched by means of the Blahut Algorithm [17]. Also, included are the Gish-Pierce asymptotic results.

In Fig.'s 4-7, the solid curves illustrating the performance of uniform threshold quantizers are obtained by successively varying the normalized stepsize δ of the quantizer.

Careful observation of these results reveals that in all cases, for a fixed number of levels N, the performance of the optimum and uniform threshold quantizers at low rates are very close. At higher rates (close to $H_{N,LM}$), however, the optimum quantizer outperforms its uniform threshold counterpart. This difference in performance is more pronounced for more broad-tailed distributions and for large values of N. In particular, by scrutinizing Fig.'s 4-7, one could easily see that for smaller values of the parameter α (more broad-tailed) the performance difference becomes more noticeable. The discrepancy between the performance of the two types of quantizers will be minimal, however, when the constraint on the number of levels is removed. In other words, the lower envelope of the performance curves of optimum quantizers approximately coincides with that of uniform threshold quantizers, i.e., $D_Q(H_0)\cong D_Q^*(H_0)$.

Empirical evidence reveals that the threshold levels of the optimum quantizer are not necessarily uniformly spaced. Moreover, on many occasions (especially for large N), we have obtained more than one quantizer achieving the optimum point, some of which are even asymmetric.

As established previously by Gish and Pierce [9], it can be seen from Fig's 4-7, that at high bit rates (low distortions) there is a constant difference (0.255 bits/sample) between the rates of the optimum quantizer and that given by the rate-distortion function for fixed values of distortion. As one could see from the results, the rate at which the performance curves approach the Gish-Pierce asymptotic result is slower for more broad-tailed densities. Moreover, our results exhibit the fact that, except possibly for the uniform distribution, the difference between the performance of optimum quantizers and the rate-distortion function becomes even smaller (in most cases in a monotonic fashion) at lower rates. This observation, however, is in sharp contrast with the results previously reported by Noll and Zelinski [10]. They concluded, on the basis of their numerical results, that the gap between the performance of optimum quantizers and the rate-distortion function becomes wider at low rates. We believe that their results are misleading mainly because of their limitation on symmetric quantizers with an even number of levels. In Table I, we have presented the correct values for the rate-distortion performance of optimum quantizers for the Gaussian and Laplacian sources, as well as the corresponding values of the rate-distortion function.

By comparing the results for different parameters, one could identify the general trend that as the p.d.f. tends to uniformity the quantizer performance at low bit rates degrades (as we have seen before for the case of uniform density), while for more peaked p.d.f.'s the performance curve remains close to the optimal curve R(D) for a wider range of distortions (see for instance, Fig. 4). In no case and for no distortion has the rate of the optimum quantizer been more than 0.3 bits/sample above the rate-distortion bound.

VII. SUMMARY AND CONCLUSIONS

We have studied the design of optimum quantizers with entropy constraint by deriving necessary conditions for optimality and proposing a recursive algorithm for solving them. We have shown that for a fixed number of quantization levels there could be a large difference in performance between the optimum and the uniform threshold quantizers. When the constraint on the number of quantization levels is omitted, however, the optimum quantizer will be only negligibly superior to the uniform threshold quantizer. In light of this fact, therefore, one could forego the formidable task of designing optimum quantizers by using a uniform threshold quantization scheme (probably with a larger number of levels) without losing much in performance. Furthermore, we have demonstrated that the difference between the performance curves of the optimum quantizer and the rate-distortion function lower bound is only 0.255 bits/sample at high rates and becomes even smaller at low rates, except for nearly uniform distributions. This is in marked contrast with previously reported results [10].

REFERENCES

1. S. P. Lloyd, "Least Squares Quantization in PCM", IEEE Trans. Inform. Theory, Vol. IT-28, pp. 129-137, March 1982.
2. J. Max, "Quantization for Minimum Distortion", IEEE Trans.Inform. Theory, Vol. IT-6, pp. 7-12, March 1960.
3. P.E. Fleischer, "Sufficient Conditions for Achieving Minimum Distortion in a Quantizer", IEEE Int. Conv. Rec., 1964, Part 1, pp. 104-111.
4. T. Berger, "Optimum Quantizers and Permutation Codes", IEEE Trans. Inform. Theory, Vol. IT-18, pp. 759-765, Nov. 1972.
5. A.N. Netravali and R. Saigal, "Optimum Quantization Design Using a Fixed-Point Algorithm", Bell Syst. Tech. J., Vol. 55, No. 9, pp. 1423-1435, Nov. 1976.
6. D.K. Sharma, "Design of Absolutely Optimal Quantizers for a Wide Class of Distortion Measures", IEEE Trans. Inform. Theory, Vol. IT-24, pp.693-702, Nov. 1978.
7. T. Berger, Rate Distortion Theory, Englewood Cliffs, NJ, Prentice Hall, 1971.
8. T. J. Goblick and J. L. Holsinger, "Analog Source Digitization: A Comparison of Theory and Practice", IEEE Trans. Inform. Theory, Vol. IT-13, pp. 323-326, April 1967.
9. H. Gish and J.N. Pierce, "Asymptotically Efficient Quantizing", IEEE Trans. Inform. Theory, Vol. IT-14, pp. 676-683, Sept. 1968.
10. P. Noll and R. Zelinski, "Bounds on Quantizer Performance in the Low Bit-Rate Region", IEEE Trans. Commun., Vol. COM-26, pp.300-304, Feb.1978.
11. M.S. Bazaraa and C.M. Shetty, Nonlinear Programming, John Wiley & Sons, Inc., New York, 1979.
12. J. W. Modestino, "Adaptive Nonparametric Detection Techniques" in Nonparametric Methods in Communications, P. Papantoni-Kazakos & D. Kazakos, pp. 29-65, New York, Marcel Dekker, 1977.
13. N. Farvardin and J. W. Modestino, "Optimum Quantizer Performance for a Class of Non-Gaussian Memoryless Sources", submitted to IEEE Trans. Inform. Theory.
14. T. Berger, "Minimum Entropy Quantizers and Permutation Codes", IEEE Trans. Inform. Theory, Vol. IT-28, pp. 149-157, March 1982.
15. V. R. Algazi and R. M. Lerner, "Binary Detection in White Non-Gaussian Noise", M.I.T. Lincoln Lab., Lexington, MA., Tech. Rept. DS-2138.
16. R. C. Wood, "On Optimum Quantization", IEEE Trans. Inform. Theory, Vol. IT-15, pp. 248-252, March 1969.
17. R. E. Blahut, "Computation of Channel Capacity and Rate-Distortion Functions", IEEE Trans. Inform. Theory, Vol. IT-18, pp. 460-473, July 1972.

Rate	Gaussian Source			Laplacian Source		
(bits/sample)	R(D)	Optimum†	Lloyd-Max	R(D)	Optimum†	Lloyd-Max
1	6.02	4.64 (4.40)	4.40	6.62	5.76 (3.01)	3.01
2	12.04	10.55(10.51)	9.30	12.66	11.31(10.85)	7.53
3	18.06	16.56(16.53)	14.62	18.68	17.20(17.09)	12.61

† Quantities in parenthesis are taken from [10]

Table 1

Comparison of the SNR (in dB) of the Optimum and Lloyd-Max Quantizer vs. the Rate-Distortion Bound for Gaussian and Laplacian Sources.

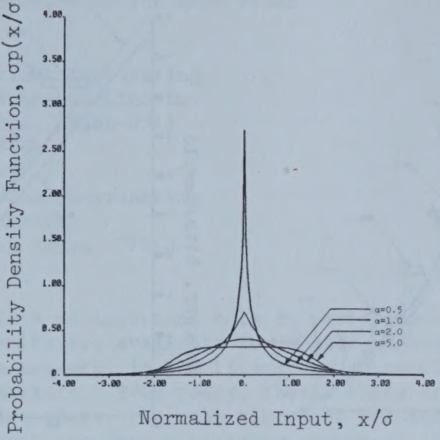


Figure 1

Probability Density Function for Generalized Gaussian Distribution with Selected Parameter Values.

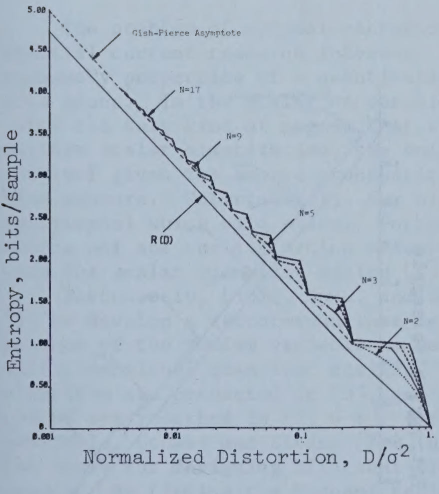


Figure 2

Performance of Optimum N-level Quantizer for a Memoryless Source with Uniform Distribution.

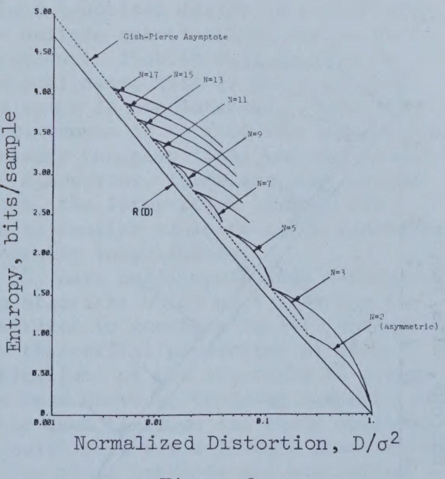


Figure 3

Performance of N-level Symmetric Uniform Threshold Quantizer for a Memoryless Source with Uniform Distribution.

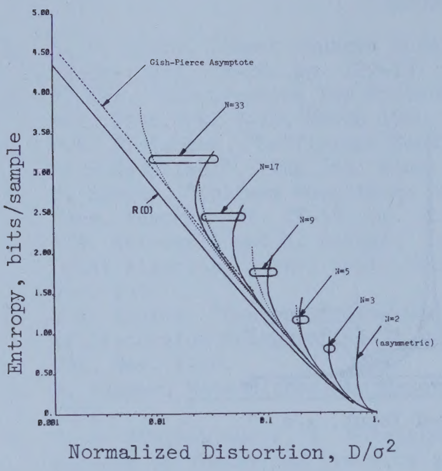


Figure 4

Performance of the Optimum and Symmetric Uniform Threshold Quantizers for a Memoryless Source with Generalized Gaussian Distribution; $\alpha=0.5$.

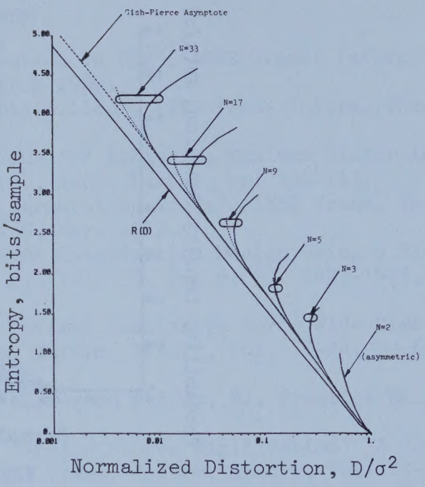


Figure 5

Performance of the Optimum and Symmetric Uniform Threshold Quantizers for a Memoryless Source with Generalized Gaussian Distribution; $\alpha=1.0$ (Laplacian).

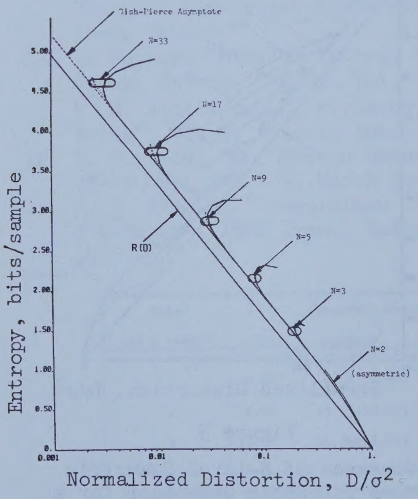


Figure 6

Performance of the Optimum and Symmetric Uniform Threshold Quantizer for a Memoryless Source with Generalized Gaussian Distribution; $\alpha=2.0$(Gaussian).

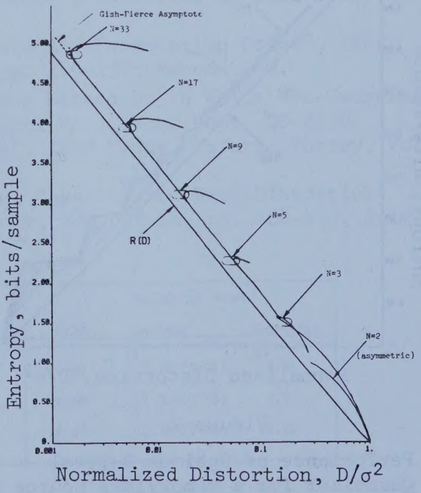


Figure 7

Performance of the Optimum and Symmetric Uniform Threshold Quantizers for a Memoryless Source with Generalized Gaussian Distribution; $\alpha=5.0$.

AN ALGORITHM FOR DESIGNING VECTOR QUANTIZERS

KHALID SAYOOD
Dept. of Electrical Engineering
University of Nebraska-Lincoln
Lincoln, Nebraska 68588-0511

JERRY D. GIBSON
Dept. of Electrical Engineering
Texas A&M University
College Station, Texas 77843

ABSTRACT

Quantization plays an important role in most communication schemes, yet, though extensive results are available for scalar quantization, until recently very little appeared in the literature concerning the design of block or vector quantizers. Even today, though there are good quantizers known for various dimensions, the only algorithm for designing vector quantizers is one which requires a training sequence. The objective of this work is to present an algorithm for designing uniform vector quantizers for any given block length without utilizing a training sequence. The algorithm is motivated by a separation technique which has appeared in the literature, and uses lattices known as the main form of Voronoi of the first kind. Vector quantizers for block lengths up to nine are generated and are seen to compare favorably with the best available quantizers in the literature.

I. INTRODUCTION

The problem of optimal vector or block quantizer design is one of substantial current research interest. For uniform quantization, one of the necessary properties of a quantization region is that it must be able to tile space. In the scalar or one-dimensional case, this is not a problem since the only kind of region that tiles space is the interval. Thus, for uniform scalar quantization, the only requirement is to find the size of the interval given the source probability density function (pdf) and the distortion measure. Unfortunately, for higher dimensions, there are many shapes (polytopes) which tile space. Furthermore, the interaction between the source pdf and the distortion measure is so complex as to make the approach used for scalar quantizer design [1] generally inapplicable.

Fortunately, Linde, Buzo, and Gray [2] have built upon Lloyd's Method I [1] to develop a vector quantizer design algorithm which uses training sequences of the random variable to be quantized to generate locally optimal multidimensional quantizer designs. The theoretical properties of this algorithm are presented in [3], and applications of the algorithm to speech coding are reported in [2, 4-6]. For the case where no training sequence is available, Conway and Sloane [7,8] use the root lattices and their duals as the basis for designing efficient block quantizers for uniformly distributed inputs. By finding the Voronoi regions for these lattices and evaluating the mean squared error (MSE) when these lattices are used to quantize uniformly distributed data, they are able to list the lattices which achieve the minimum MSE for dimensions one through ten. These results constitute the best known uniform vector quantizers for the given block-lengths. Additional important results on vector quantization are available in the very readable paper by Gersho [9] and the special issue of the IEEE TRANSACTIONS ON INFORMATION THEORY edited by Gray [10].

In his excellent paper, Gersho [9] states, "What is needed is a theory of quantization for finite block lengths of arbitrary size." While we do not purport to present a theory of quantization, we do present an algorithm for designing uniform vector quantizers for finite block lengths of arbitrary size,which does not require a training sequence. The distortion measure is the familiar per letter MSE distortion also considered in [7-9].

The quantizer design algorithm presented here is motivated by Sakrison's result [11] that the distortion measure determines the shape of the optimum quantization region while the source probability density function (pdf) determines the optimum encoding of the output points. This result is valid only for small distortion and as the blocklength or number of dimensions becomes large. Thus, although we cannot prove optimality for finite blocklengths, we are led to pursue the design of vector quantizers for uniformly distributed inputs. The vector quantization problem therefore reduces to finding that quantization region which minimizes the squared error distortion measure.

Of course, we have now returned to the problem mentioned earlier of finding the "best" quantization region out of many which tile k dimensional space. The vector quantizer design algorithm presented here finds the best quantization region by constructing lattices from the vectors of a regular Zelling set and then generating Dirichlet partitions between adjacent lattice points. The performance of the quantizers so designed is evaluated by computing the mean squared quantization error per symbol for dimensions 1-9 using a Monte Carlo integration method. To determine the performance of these quantizers for a clearly nonasymptotic case and for nonuniformly distributed inputs, a four dimensional quantizer is used to quantize Laplacian and gamma distributed random variables at 1 and 2 bits/sample. The results are compared to the rate distortion bound and to scalar quantization.

II. NOTATION

Let X be a k dimensional source vector with joint pdf $p(x) = p(x_1, x_2, \dots, x_k)$. A k dimensional vector quantizer is a function $Q(x)$ which maps $x \in R^k$ into one of N output points with each output point corresponding to an output vector $Y_1, Y_2, \dots Y_N$, belonging to R^k. The quantizer is completely specified by listing the N output vectors and their corresponding partitions of the space R^k into N disjoint and exhaustive regions. Following [2] and [9] let these regions be denoted by $\mathcal{S}_1, \mathcal{S}_2, \dots \mathcal{S}_N$ so that

$$Q(x) = Y_i, \text{ if } X \in \mathcal{S}_i, \text{ for } i = 1,2,\dots N.$$

Such a vector quantizer may also be called a block quantizer with blocklength k. Note that if k=1, we have the familiar scalar quantization problem.

The distortion measure selected for this work is the mean squared error (MSE) per symbol (letter) given by

$$\begin{aligned} d &= \frac{1}{k} E\{d(x-y)\} \\ &= \frac{1}{k} E \,||X-Q(x)||^2 \\ &= \frac{1}{k} \int_{R^k} ||X-Q(x)||^2 p(x)\, dx \qquad (1) \end{aligned}$$

where $||\cdot||$ denotes the usual ℓ_2 norm.

III. MOTIVATION

For optimal nonuniform vector quantizer design, it is desired to find a quantizer which minimizes (1). Due to the interaction of the source pdf and the error measure $X-Q(x)$, finding such quantizers is quite difficult. A great simplification would be obtained if somehow the source pdf could be ignored while the optimum quantization region is constructed. Sakrison [11] demonstrates that such an approach is valid for the asymptotic case of a large number of dimensions (k large). It is worthwhile to briefly consider Sakrison's development.

The average mutual information between the quantizer input x and the output y is given by

$$I(X;Y) = H(X) - H(X|Y). \tag{2}$$

Letting

$$(G(d^*) = \{p(y|x) : d \leq d^*\} \tag{3}$$

where p(y|x) represents the transition probabilities, and noting that d in (1) is a difference distortion measure, i.e.,

$$d = E\{ d (X-Y) \},$$

we can write for the rate distortion function

$$R((d^*) = \min_{f(y|x) \in G(d^*)} I(X;Y)$$

$$\geq H(X) - \max_{f(y|x) \varepsilon G(d^*)} H(X-Y). \tag{4}$$

Defining Z = X-Y, we can rewrite (4) as

$$R(d^*) \geq H(X) - \max_{E[d(Z)] \leq d^*} H(Z), \tag{5}$$

where the right hand side of (5) is known as the Shannon lower bound. Equality in the above expression holds when the random variable X can be expressed as the sum of two independent random variables, one of which is distributed according to the probability density

$$f(Z) = \frac{e^{sd(z)}}{\int dz e^{sd(z)}} \tag{6}$$

where s is a negative constant [12, Thm. 4.3.1]. Equality is also approached for a wide class of distributions when d* is small (See [6], Thm. 4.3.4 and Thm. 4.3.5 . Thus, for d* small, the maximization on the right side of (5) depends only on the distortion criterion. Sakrison further shows that the maximum is achieved when the quantization regions are of the form

$$u = \{\chi \varepsilon R^k : d(X) \leq d^*\}. \tag{7}$$

Therefore, for d* small, the distortion criterion alone determines the shape of the quantization region.

For the asymptotic case of a large number of dimensions, this result not only provides a means for separating the effects of the pdf and the distortion criterion, it also delineates the shape of the ideal quantization region. That the ideal quantization region should be of this form is also pointed out by Gish and Pierce [13]. They also provide an expression for the approximate average error. Letting

$$M_u (V) = \frac{1}{V_u} \int d (X - X^0) \, dX \qquad (8)$$

where X^0 is the value that minimizes the integral and V is the volume of the region Then, the average error is approximately given by

$$E \cong M_u (V). \qquad (9)$$

The problem with applying this asymptotic result to the more practical nonasymptotic case is that for the ℓ_2 norm distortion measure, the ideal regions are hyperspheres which cannot cover space without overlap. Note that hyperspheres arise not because they are selected for their excellent covering properties (which, in fact, they do not possess [14]) but because they are dictated by the distortion criterion.

Heuristically, a compromise to using hyperspheres as the quantization region would be to approximate this ideal shape with a shape which tiles space. However, the question remains as to what sense the approximation should be taken. Equation (8) provides a partial answer to the question. Note that for d* small, M_u (V), and hence the average error, is minimized when the volume V is maximized. Hence, the approximation should be one which has maximum volume over all space-tiling regions which fit inside (7). This, as stated before, is only a partial answer. However, there are other properties of uniform vector quantizers which have not been utilized as yet.

Gersho [9] points out that for optimal quantization the partitions between output points are Dirichlet (nearest neighbor) partitions. A Dirichlet partition is defined as follows:

Dirichlet Partition: For any finite or countably infinite set of points $X_1, X_2, \ldots X_N$, $X_i \in R^k$, i = 1, 2,...,N, a dirichlet partition P_i is given by

$$P_i = \{X \in R^k | \; || X - X_i || \leq || X - X_j ||, j \neq i\} \qquad (10)$$

Gersho also notes that the output points should be centroids of the quantization regions.

Necessary conditions for a quantization region can therefore be listed as follows:

1. It should tile space
2. The partitions bounding it should be Dirichlet partitions
3. The output points should be centroids of the quantization regions

Add to these the heuristically obtained condition

4. The quantization region should approximate the shape (7) in the sense previously stated

For uniform quantizers a further condition can be added

5. Because of the regular structure of the uniform quantizer the output points should form a lattice.

A lattice is defined as follows:

Lattice: Let $A_1, A_2, \ldots, A_k$ be a set of basis vectors for R^k. Then the set

$$L = \{ X : X = u_1 A_1 + u_2 A_2 + \ldots + u_k A_k, \quad u_i \in \mathbb{Z}, \; i = 1, \ldots, k\} \qquad (11)$$

is called a lattice where $\mathbb{Z}$ denotes the set of integers.

It should be noted that while conditions 1-3 are necessary conditions they may not be sufficient. Condition 4 is a heuristically obtained condition motivated by the asymptotic case.

With this caveat in mind, a solution which satisfies all these conditions is a least dense covering lattice. A least dense covering lattice is a space filling lattice in which the lattice points are centers of spheres which cover space with minimum overlap [14]. The quantization regions are formed by constructing Dirichlet partitions between neighboring points. This effectively divides the overlap region among the neighboring points. As this is a minimal overlap covering,the volume of the region is maximized and condition 4 is satisfied. Because of the periodicity of the lattice the regions thus constructed are parallelohedra with centroids at the center of the circumscribing sphere thus satisfying condition 3. The other conditions are obviously satisfied.

IV. QUANTIZER DESIGN

While a least dense covering lattice seems to be the answer to the problem of vector quantizer design, unfortunately, the least dense covering lattices are not known for all dimensions. However, there do exist lattices which have been shown to be optimal for lower dimensions and at least locally optimal for all dimensions. These lattices are based on regular Zelling set **(Selling frame)**.

Zelling set: A Zelling set Z in R^k is a set of k + 1 vectors with the property that any vectors of the set provide a basis for the lattices, and if

$$Z = \{X_1, X_2, \ldots, X_{k+1}\} \tag{12}$$

then

$$X_1 + X_2 + \ldots + X_{k+1} = 0 \tag{13}$$

The dot products

$$g_{ij} = X_i \cdot X_j \qquad i \neq j \tag{14}$$

are known as the Zelling parameters of the set.

Regular Zelling set: A Zelling set in which all the Zelling parameters are identical is known as the regular Zelling set.

A lattice constructed on arbitrary vectors from a regular Zelling set is known as the main form of Voronoi of the first kind and is denoted by L_1^k where k denotes the dimensionality of the Euclidean space being covered.

Theorem: To lattice L_1^k there corresponds a local minimum of the density of lattice coverings of R^k.

The proof is to be found in Gameckii [15]. For dimensions two through five these lattices have been shown to be optimum [16-20]. However, for large k, Davenport [21] and Watson [22] have proven the existence of lattices which have density less than L_1^k. To the authors' knowledge, no such lattice has actually been constructed.

To generate a lattice in R^k, k basis vectors are needed.Counting each component as an unknown there are k^2 unknowns. Using the properties of the regular Zelling set and fixing one of the k + 1 vectors to be

$$X_0 = [1\ 0\ .\ .\ .\ 0] \tag{15}$$

the following equations are obtained:

N equations are obtained from the fact that

$$X_0 + X_1 + \ .\ .\ .\ + X_k = 0 \tag{16}$$

Letting the Zelling parameter be g, $k(k+1)/2$ equations are obtained from the fact that the scalar products are all equal to g. Notice that as g is unknown there are now $k^2 + 1$ unknowns. However, only $k+k(k+1)/2$ equations are available, where

$$n + \frac{n(n+1)}{2} < n^2 + 1 \text{ for } n \geq 3.$$

The way out of this quandary is to realize that the orientation of the lattice is of no concern as the distortion regions are spherical. Using this fact, some of the unknowns can be eliminated. Taking the case of R^3, let the basis vectors be

$$\begin{aligned} X_1 &= [x_{11}\ x_{12}\ x_{13}] \\ X_2 &= [x_{21}\ x_{22}\ x_{23}] \\ X_3 &= [x_{31}\ x_{32}\ x_{33}] \end{aligned} \tag{17}$$

X_1 can be fixed in the xy plane be setting x_{13} to zero. The component x_{13} could have been set to any other constant, but setting it to zero simplifies the equation. Now there are $3^2 + 1 - 1 = 9$ unknowns and $3 + 3(4)/2 = 9$ equations. Similarly in R^4, x_1 is fixed in the xy plane by setting x_{13} and x_{14} to zero and x_2 is fixed in the xyz hyperplane by setting x_{24} to zero, thus there are $4^2 + 1 - (1+2)$ unknowns and $4 + 4(5)/2 = 14$ equations. This reasoning can be extended to R^k and the number of unknowns becomes

$$\text{No. of unknowns} = k^2 + 1 - \sum_{k=1}^{k-2} k \tag{18}$$

$$= k^2 + 1 - \frac{(k-2)(k-1)}{2} \tag{19}$$

$$= \frac{k^2}{2} + \frac{3}{2} = k + k\ (k + 1)/2 \tag{20}$$

which is the same as the number of equations. From now on for the sake of brevity define

$$M = k + k(k + 1)/2 \tag{21}$$

Now there are M equations and M unknowns, but a problem still remains. Only 2N of these equations are linear, obtained from summing the components and from inner products with X_0. The other M - 2 equations are all nonlinear.

Because of the presence of the nonlinear equations, a numerical method is needed. The Newton Raphson method has been found to be eminently suitable for this problem, taking less than ten iterations to converge to the solution. The tolerance level for the norm square of the error was 10^{-14}. The initial condition for the basis vectors was the canonical basis. Where the canonical basis vector X_j has a one as its$(j-1)$th element, and all the other elements are zero.

Having obtained the basis set for the lattice, it is a straightforward matter to obtain the rest of the points in the lattice. To obtain the quantization region, Dirichlet partitions are constructed between neighboring points.

Notice that until now no mention has been made of the distortion sphere which circumscribes the quantization region. The lattice constructed is a covering of space with spheres of some fixed radius which might not (and will probably not) be d*. The last step therefore is to adjust the lattice to correspond to this constraint. To do this define the density of the covering lattice $\theta\ (k)$ as

$$\theta(K) = \frac{V(K)}{\det(L)} \qquad (22)$$

where K is a hypersphere, V(K) is the volume and det(L) is the absolute value of the determinant of the matrix of basis vectors of L. For the main form of Voronoi of the first kind

$$\theta(L_1{}^k) = (1.17)^k \qquad (23)$$

Then

$$\theta(L_1{}^k) = (1.17)^k = \frac{V(K)}{\det\ (L_1^k\ (K))} \qquad (24)$$

where K is a hypersphere in R^k of radius d^*.

Therefore (from [23])

$$V(K) = \frac{\pi^{k/2}\ d*^k}{\Gamma\ (\frac{k}{2} + 1)} \qquad (25)$$

or

$$V(K) = \begin{cases} \pi^M\ d*^{2M})/M! & ,\ k = 2M \\ \dfrac{\pi^M\ M!\ (2d*)^{2M+1}}{(2M + 1)!} & ,\ k = 2M + 1 \end{cases} \qquad (26)$$

so for covering with hyperspheres of radius d* the determinant $D(L_1^k(K))$ should be

$$\det(L_1^k(K)) = \frac{V(K)}{\theta(L_1^k)} = \frac{V(K)}{(1.17)^k} \qquad (27)$$

Call this value D*. Call the value of the determinant of the lattice actually obtained, D, and let

$$D^* = \alpha D \qquad (28)$$

then to adjust the lattice each component of the basis set needs to be multiplied by $(\alpha)^{1/k}$ to obtain the desired lattice.

V. EVALUATION OF DESIGN

The foundation for this design technique was laid down using Sakrison's [11] result that the pdf of the source could be ignored in obtaining the shapes of the quantization regions. It seems reasonable to expect, therefore, that there is a method for evaluating the design independent of the source pdf. In fact such a measure was introduced by Gersho. He called it the coefficient of quantization. This was shown to be the mean square quantization error per symbol by Conway and Sloane.

The mean square quantization error per symbol (MSQE) is given by

$$I(H) = {}_H\!\int\ ||X - \hat{X}||^2\ dX\ /\ kV(H)^{1+2/k} \qquad (35)$$

where H is the quantization region, $\hat{X}$ is the centroid of H, V(H) the volume of H, and k is the dimensionality.

As the quantization regions are translated versions of the region H, for computational ease, H can be taken as the quantization region containing the origin. Then $\hat{X}$ is zero and the equation (35) somewhat simplified. However, the integral is over the quantization region, and as these regions have a non-standard shape, standard integration techniques, analytical or numerical, are difficult to use. Therefore to be able to obtain an evaluation of the design, crude Monte-Carlo integration was used. An introduction and development of these techniques can be found in [24] and [25]. The variance of the estimates were found by [24]

$$\hat{\sigma}^2 = \frac{1}{N} \sum_{i=1}^{N} y_i^2 - (\frac{1}{N} \sum_{i=1}^{N} y_i^2)$$

where N is the number of samples used and y_i is the value of the integrand as a function of the pseudo-random number x_i. By using an appeal to the central limit theorem, Hammersley and Handscomb [25] show that the 95% confidence interval for estimate I is given by

$$\hat{I} - 2\hat{\sigma}/\sqrt{N} \leq \hat{I} \leq \hat{I} + 2\hat{\sigma}/\sqrt{N}$$

As a million samples were used and the variance of the estimates were on the order of 10^{-4} the resulting confidence intervals were $\hat{I} \pm 10^{-5}$. The results are tabulated alongside the results for the best known quantization region in Table I. The latter results were obtained from Conway and Sloane [7].

Table I Vector Quantizer Performance Comparison
Mean Square Quantization Error per Symbol

Dimensionality	Best Available	Present Design
1	.0833	.0833
2	.0802	.0802
3	.0785	.0785
4	.0766	.0766
5	.0756	.0734
6	.0743	.0727
7	.0732	.0723
8	.0717	.0715
9	.0747	.0704

Note that the results for the present design are at least as good as the best available. The significance of this is the fact that all these quantizers were obtained using a design procedure. This is in contrast to the other designs where the best available quantizers were obtained by searching through various possibilities.

For a four-dimensional quantizer, signal to noise ratios for low bit rates were obtained via simulation and compared to signal to noise ratios for optimum scalar quantizations and the rate distortion bound. The latter figures were obtained from Noll and Zelinski [26]. These simulations were performed using 50,000 samples from a pseudorandom number generator. The size of the quantization region was adjusted empirically to provide the best result.

Table 2 Signal to Noise Ratio Comparison

Distribution	Rate	Signal to Noise Ratios				
	bits/ sample	RDF	Vector (Ent)	Scalar (Ent)	Vector (Uncod)	Scalar (Uncod)
Laplacian	1	6.62	4.92	3.01	3.97	3.01
	2	12.66	12.25	10.85	9.81	7.53
Gamma	1	8.53	8.35	1.77	4.15	1.77
	2	15.21	13.79	11.60	7.55	6.33

In the table Ent implies the outputs were entropy coded and uncod means they were not. The figures under RDF are the values of the rate distortion bound. Notice that the use of vector quantization brings the output entropy of the quantizer closer to the rate distortion bound. This is especially evident in the two bit Laplacian and the one bit Gamma case.

VI. CONCLUSIONS

In this paper an algorithm for designing vector quantizers for any blocklengths was presented. The design entailed using asymptotic results, for nonasymptotic cases. The justification for this approach is the results obtained by utilizing it. As can be seen from Table 1, the relatively simple design procedure results in designs which compare favorably with the best available vector quantizers. Table 2 is a strong argument for using vector quantizers in low bit rate situations. It should be noted that the results for vector quantizers shown in Table 2 are not the best possible. They are presented so as to give some idea of the improvements possible through vector quantization.

REFERENCES

[1] Lloyd, IEEE Trans. Inform. Theory, vol. IT-28, pp. 129-137, March 1982.

[2] Linde et. al., IEEE Trans. Commun., vol. COM-28, pp.84-95, Jan. 1980.

[3] Gray et.al., Information and Control, vol. 45, pp.178-198, 1980.

[4] Buzo et.al., IEEE Trans. Acoust., Speech, and Signal Processing, vol. ASSP-28, pp.562-574, Oct. 1980.

[5] Rebolledo et.al., IEEE Trans. Commun., vol. COM 30, pp.721-727, April 1982

[6] Abut et.al., IEEE Trans. Acoust., Speech, and Signal Processing, vol. ASSP-30, pp.423-435, June 1982.

[7] Conway and Sloane, IEEE Trans. Inform. Theory, vol. IT-28, pp.211-226, March 1982.

[8] Conway and Sloane, IEEE Trans. Inform. Theory, vol. IT-28, pp.227-232, March 1982.

[9] Gersho, IEEE Trans. Inform. Theory, vol. IT-25, pp.373-380, July 1979

[10] Gray, ed., IEEE Trans. Inform. Theory, Part I, vol. IT-28, March 1982

[11] Sakrison, in Image Transmission Techniques, Pratt, ed., New York: Academic Press, 1979, pp.21-71.

[12] Berger, Rate Distortion Theory, Englewood Cliffs: Prentice-Hall, 1971.

[13] Gish and Pierce, IEEE Trans. Inform. Theory, vol. IT-14, pp.676-683, Sept. 1968.

[14] Rogers, Packing and Covering, Cambridge: Cambridge University Press, 1964.

[15] Gameckii, Soviet Math. Dokl., vol. 3, pp.1410-1414, 1962.

[16] Kershner, Amer. J. of Math., vol. 20, pp.25-52, 1954.

[17] Bambach, Proc. Indian Nat. Science Academy, vol. 20, pp.25-52, 1954.

[18] Bambach, Camb. Phil. Soc. Proc., vol. 50, pp.203-208, 1954

[19] Delone and Ryskov, Soviet Math. Dokl., vol. 4, pp.1333-1334, 1963.

[20] Ryskov and Barnakovski, Soviet Math. Dokl., vol. 16, pp.586-590, 1975.

[21] Davenport, Rend. Circ. Math. Palermo, Series II, vol. 1, pp.92-107, 1955.

[22] Watson, Rend. Circ. Math. Palermo, Series II, Vol.5, pp.93-100, 1956.

[23] Somerville, An Introduction to the Geometry of N Dimensions, New York: Dover, 1958.

[24] Schrieder, in The Monte-Carlo Method, Schrieder, ed., pp.91-136, 1967.

[25] Hammersley and Handscomb, Monte-Carlo Methods, London: Methuen, 1965.

[26] Noll and Zelinsky, IEEE. Trans. Commun., vol.COM-26, pp.300-304, Feb. 1978.

SPHERICAL COORDINATES QUANTIZERS:
ASYMPTOTIC ANALYSIS AND THE THREE-DIMENSIONAL CASE

Peter F. Swaszek

Department of Electrical Engineering
University of Washington
Seattle, Washington 98195

Abstract

Spherical coordinates quantizers for multidimensional spherically symmetric sources are designed for the Mean Square Error criterion when the number of quantization levels is large, i.e. fine quantization. These parallel scalar processors, while they do not perform as well as optimal quantizers for the same source, have a simple implementation. Their performance for a three-dimensional source with generalized Gaussian marginals is compared to the performance of optimal and rectangular coordinate quantizers.

1. Introduction

The use of polar quantizers for bivariate, circularly symmetric random sources is well established in the technical literature [1,2,3,4]. After effecting a change from rectangular to polar coordinates, the resulting magnitude and phase are quantized using separate scalar quantizers. This author [5] has previously considered the extension of the quantizers mentioned above to the quantization of a k-dimensional Gaussian random source. This paper continues that discussion for spherically symmetric sources when it is assumed that the number of quantization levels, N, is large. The particular case where $k=3$ will also receive close attention. Quantizers for these trivariate, spherically symmetric sources are examined for the following two reasons:

1- Actual sources which display spherical symmetry can be found in our three dimensional physical world. For example, the density of particles given off by a point source, etc.

2- If $f_x(\,)$ is the marginal density of the source then the three dimensional spherically symmetric density has the following simple form:

$$f(r,\varphi_1,\varphi_2) = [\, -2\, r\, f_x{}'(r)\,] \times \frac{1}{2\pi} \times \frac{\cos \varphi_2}{2}$$

We consider a data vector $\mathbf{x}$ from a spherically symmetric source with rectangular coordinate elements x_j, $j=1,2,\ldots k$. The source's statistics are contained in its k-dimensional density function $f(\mathbf{x})$. Using a transformation to k-dimensional spherical coordinates, the resulting magnitude and $k-1$ angles will be separately quantized. The magnitude and angles quantizers are derived when *Mean Square Error* (MSE) is the performance criterion and the number of levels is large. Examples and comments conclude the paper.

2. Spherically Symmetric Densities

Spherically symmetric sources are characterized by contours of constant height that are hyperspheres in the k dimensional space. These spherically symmetric densities can be generated by replacing the independent variable of a zero mean, unit power univariate density, say $p(x)$, with the square root of a quadratic form [6,7]. The resulting multivariate density has the form:

$$f(\mathbf{x}) = \gamma p\left[\sqrt{\mathbf{x}^T\mathbf{x}}\right] \ ; \ \mathbf{x} \in \mathbf{R}^k$$

where γ is a scaling constant so that the density has unit mass. The resulting multivariate density, however, does not in general have as its marginals the univariate $p(x)$. There is a method of generating spherically symmetric densities with a specific marginal. Assume the marginal has characteristic function $\Phi_1(u)$, and let the k-dimensional characteristic function be $\Phi_k(\mathbf{u})=\Phi_1(\sqrt{\mathbf{u}^T\mathbf{u}})$ Taking the inverse transform yields the k-dimensional density; care must be taken to ensure that it integrates to one and is positive for all $\mathbf{x}$.

A useful set of coordinates with which to describe an input from a spherically symmetric source is the magnitude r and $k-1$ angles φ_j's of the k-dimensional spherical coordinates system [8]. The following transformations produce these coordinates:

$$r = \left[\sum_{j=1}^{k} x_j^2\right]^{1/2} \ ; \ \varphi_j = \tan^{-1}\left[x_{j+1} \Big/ \left[\sum_{i=1}^{j} x_i^2\right]^{1/2}\right] \ , j = 1,2,\ldots k-1$$

A change of variables produces the spherical coordinate source density

$$f(\mathbf{x}) \rightarrow f(r,\underline{\varphi}) = f_k(r) \prod_{j=1}^{k-1} f_j(\varphi_j)$$

where

$$f_k(r) = \frac{2\,\pi^{k/2}}{\Gamma(k/2)} r^{k-1}\ \gamma p(r) \quad , \ r \in [0,\infty) \quad ; \quad f_1(\varphi_1) = \frac{1}{2\pi} \quad , \ \varphi_1 \in [0,2\pi)$$

and

$$f_j(\varphi_j) = \frac{\Gamma[(j+1)/2]}{\Gamma(1/2)\Gamma(j/2)} \cos^{j-1}\varphi_j \quad ; \ \varphi_j \in [-\pi/2,\pi/2] \ , j = 2,3,\ldots k-1$$

The resulting spherical coordinates are statistically independent.

In the spherical coordinates representation, employing separate scalar quantizers defines the typical quantization region as the intersection of a nonzero width spherical shell centered at zero with a pyramid of apex zero. The solution outlined in this paper involves the design of these scalar quantizers and the allocation of the number of levels to each.

3. Quantizer Optimization

The quantizers designed in this paper, called Q_r and Q_1 through Q_{k-1} for the magnitude and $k-1$ angles respectively, are all scalar processors. For a scalar input s, a quantizer's output is $\hat{s}$. We assume that the number of quantization levels is large and the compandor model of Bennett [9] for a nonuniform quantizer is appropriate (see Fig. 1). Under this model, the quantizer Q_s is a three-part system: an invertible, differentiable compressor nonlinearity g_s mapping the range of the input to [0,1], a uniform quantizer Q_U with N_s levels on [0,1] and an expandor $h = g_s^{-1}$ mapping [0,1] back to the range of the input signal. For this model, specification of the compressor g_s and the number of levels N_s completely determines the quantizer. When N_s is large and g_s is smooth, the output of a compandor system for an input s can be approximated by

$$\hat{s} \approx s + \varepsilon h'[g_s(s)] = s + \frac{\varepsilon}{g_s'(s)}$$

where ε is an independent noise source uniformly distributed on $[-\Delta/2, \Delta/2]$ ($\Delta = 1/N_s$ = the step size of the quantizer Q_U). After substituting for $\hat{s}$, the calculus of variations may be employed to yield the best compressor function.

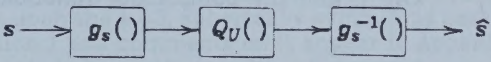


Fig. 1 - Compandor model of a quantizer.

For a quantizer with input $\mathbf{x}$ and output $\hat{\mathbf{x}}$ the MSE D is

$$D = \int_{\mathbb{R}^k} |\mathbf{x} - \hat{\mathbf{x}}|^2 f(\mathbf{x})\, d\mathbf{x}$$

In the previous paper [5], it was shown that D can be rewritten as

$$D = M_{k-1}^2 \times E_k(N_r) \quad + \quad k\left[1 - M_{k-1}^2\right] \tag{1}$$

where $E_k(N_r)$ is the MSE of the quantizer with N_r levels which is optimal for the magnitude density and M_{k-1} is defined sequentially by ($M_0 = 1$)

$$M_j = \frac{\Gamma[(j+1)/2]}{\Gamma(1/2)\Gamma(j/2)} \int_{-\pi/2}^{\pi/2} (\sin\varphi_j \sin\hat{\varphi}_j + M_{j-1}\cos\varphi_j \cos\hat{\varphi}_j) \cos^{j-1}\varphi_j \, d\varphi_j \tag{2}$$

Conditions to optimize the magnitude and angle quantizers for small N are found in [5]. Inspection of the above expressions demonstrate that minimizing E_k and maximizing each M_j over the appropriate compressor functions minimizes the overall MSE.

To minimize $E_k(N_r)$ for large N_r, we employ the minimum MSE compressor for the magnitude density

$$g_r(r) = K_r \int_0^r f_k^{1/3}(t)\, dt \tag{3}$$

where K_r is a constant such that g_r maps to [0,1]. The actual compandor system has its expandor scaled by M_{k-1}.

For large N_j we cannot immediately decide to employ the minimum MSE compressor for the angle densities since we are trying to maximize M_j in Eq.(2) for each j, not minimize the mean square error between φ_j and $\hat{\varphi}_j$. Assuming that $M_{j-1} \approx 1$ in Eq.(2) (since N_{j-1} is also large, M_{j-1} will be close to unity), the term in parenthesis simplifies to $\cos(\varphi_j - \hat{\varphi}_j)$. This is expanded in a Taylor series about zero, since for large N_j the region widths will be small, and only the first two terms are kept. Applying the compandor approximation and the calculus of variations yields the compressor function for the j-th angle:

$$g_j(\varphi_j) = K_j \int_{-\pi/2}^{\varphi_j} \cos^{(j-1)/3}\vartheta \, d\vartheta \tag{4}$$

where again K_j is a constant so that g_j maps to $[0,1]$. This resulting compressor is seen to be the minimum MSE compressor for the angle density and is proportional to an incomplete beta function. For the φ_1 quantizer, the lower limit in Eq.(4) is zero, $g_1(\varphi_1)$ is linear and the quantizer is uniform.

The overall result is that to minimize the MSE of a k-dimensional spherical coordinates quantizer, a factorization of the total number of levels N must be selected, $N = N_r \times N_1 \cdots \times N_{k-1}$, and the compressor functions are found from Eqs.(3) and (4).

4. Asymptotic Results

This section considers asymptotic MSE rates and the solution to the problem of factoring the number of levels to each of the spherical coordinates quantizers. We assume that the number of levels in each quantizer is large so that the compandor approximation is appropriate. The levels are factored to $N = N_r \times N_{\underline{\varphi}}$ where $N_{\underline{\varphi}}$ is denoted as the product of the number of levels in all of the angle quantizers. The expression for D in Eq.(1) requires expressions for the magnitude error E_k as a function of N_r and for M_{k-1} as a function of $N_{\underline{\varphi}}$. Previous asymptotic results [10] yield

$$E_k(N_r) \approx \frac{1}{12\, N_r^2} \left[\int_0^\infty f_k^{1/3}(r)\, dr \right]^3 = \frac{E_r}{N_r^2}$$

It is shown in Appendix A that M_{k-1} is of the form

$$M_{k-1} \approx 1 - C_{k-1}\, N_{\underline{\varphi}}^{-2/(k-1)}$$

with C_{k-1} a constant dependent only upon k. The solution to the factorization of the number of levels in the angle quantizers is also listed in Appendix A. From the value of C_{k-1}, Eq.(1) is minimized by

$$N_r \approx \left\{ \frac{E_r\ (k-1)}{2k\ C_{k-1}} \right\}^{(k-1)/2k} N^{1/k}$$

and the minimum spherical MSE is

$$MSE_S \approx k \left\{ \frac{2k\ C_{k-1}}{k-1} \right\}^{(k-1)/k} E_r^{1/k}\, N^{-2/k} \tag{5}$$

5. Examples

In one-dimensional or rectangular quantization, each coordinate x_j is quantized independently by a Max-type, scalar quantizer [11]. The resulting quantization regions, being the cross product of k intervals, are k-dimensional rectangular parallelepipeds. Each coordinate has a MSE term $E_1(N_j)$ which is the error associated with a scalar quantizer for the marginal density with N_j levels $(N_1 \times N_2 \cdots \times N_k = N)$. The errors, being independent and orthogonal, sum to the total error. A symmetry argument shows that this rectangular quantization error is minimized when each $N_j = N^{1/k}$. The total error is then $k \times E_1(N^{1/k})$.

Optimal multidimensional quantization of sources has been treated by Zador [12] and others. These vector input devices attain the lowest error rates possible but are often difficult to implement. The optimal MSE rate is

$$MSE_O \approx \frac{k\ C(k,2)}{N^{2/k}} \left\{ \int_{\mathbb{R}^k} f^{k/(k+2)}(\mathbf{x})\, d\mathbf{x} \right\}^{(k+2)/k}$$

where $C(k,2)$ is a constant depending upon the optimal k-dimensional uniform quantizer. Zador provided bounds on this constant. More recently, Conway and Sloane [13] found tighter upper limits on $C(k,2)$ for k between 3 and 10.

We now leave the general k arguments and for the reasons expressed in the Introduction focus upon the $k=3$, spherically symmetric case. Derivation of the expression in Section 1 and comments on it appear in Appendix B. For the rectangular case as stated above, the resulting three scalar quantizers are identical and the errors, being orthogonal, add to yield:

$$MSE_R \approx \frac{1}{4\, N^{2/3}} \left[\int_{-\infty}^{\infty} f_x^{1/3}(x)\, dx \right]^3 \tag{6}$$

The minimum achievable MSE is a function of the best three dimensional uniform quantizer and the source density. For the truncated octahedron as the uniform quantizer region, the value of $C(3,2)$ is 0.0785445 [14]. Mapping the density integral to the spherical space by $f(x,y,z) \rightarrow f(r,\varphi_1,\varphi_2)/r^2 \cos\varphi_2$ and $dx\ dy\ dz \rightarrow r^2 \cos\varphi_2\, dr\ d\varphi_1\, d\varphi_2$ and integrating yields:

$$MSE_O \approx \frac{3\,(4\pi)^{2/3}\, C(3,2)}{N^{2/3}} \left[\int_0^{\infty} [\, -2\, r\, f_x'(r)\,]^{3/5}\, r^{4/5}\, dr \right]^{5/3} \tag{7}$$

For the spherical coordinates quantizer, the MSE from Eq.(5) is

$$MSE_S \approx \frac{3^{1/3}\, \pi^{7/6}\, \Gamma(2/3)}{2^{4/3}\, \Gamma(7/6)\, N^{2/3}} \int_0^{\infty} [\, -2\, r\, f_x'(r)\,]^{1/3}\, dr \tag{8}$$

As an example, consider these error rates for a three-dimensional, spherically symmetric source with generalized Gaussian marginals [15]. This family of sources, parameterized by a constant $c>0$, has marginal density function

$$f(x) = \frac{c\,\eta(c)}{2\Gamma(1/c)}\, e^{-[\eta(c)|x|]^c} \quad ; \quad \eta(c) = \left[\frac{\Gamma(3/c)}{\Gamma(1/c)} \right]^{1/2} \quad ; \ c > 0$$

For this source model the MSE rates reduce to

$$MSE_R \approx \frac{3^{3/c}\, \Gamma^3(1/c)}{c^2\, \Gamma(3/c)}\, N^{-2/3}$$

$$MSE_S \approx \frac{3^{2/3}\, 3^{1/c}\, \pi^{7/6}\, \Gamma(2/3)}{2^{4/3}\, c^{1/3}\, \Gamma(7/6)\, \Gamma^{1/3}(3/c)}\, \Gamma\!\left(\frac{1}{c} + \frac{1}{3} \right) N^{-2/3}$$

and

$$MSE_O \approx \frac{3\, C(3,2)\, 2^{4/3}\, \pi^{2/3}\, c^{1/3}}{\Gamma(3/c)}\, \Gamma^{5/3}\!\left(\frac{3c+9}{5c} \right) \left(\frac{5}{3} \right)^{1+\frac{3}{c}} N^{-2/3}$$

Rather than directly displaying the three MSE rates as functions of the parameter c, instead we exhibit a differential *Signal-to-Noise Ratio* defined by

$$\Delta SNR_{1,2} = SNR_1 - SNR_2 = 10 \log_{10} \frac{MSE_2}{MSE_1} \, dB$$

In Fig. 2, we plot this differential *SNR* for optimal and spherical quantizers with respect to rectangular quantizers for the generalized Gaussian source.

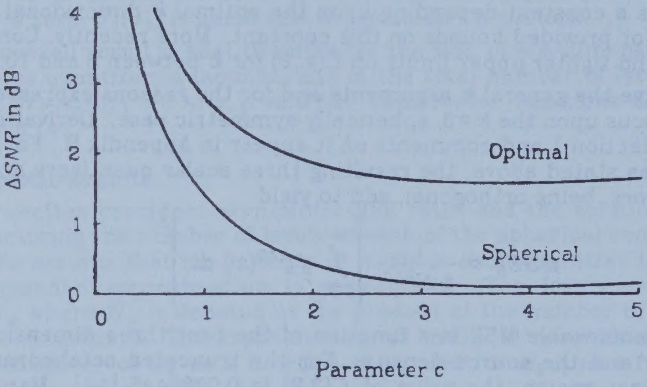


Fig. 2. Differential *SNR* for the generalized Gaussian source.

6. Discussion of Densities

In this brief section, the validity of spherically symmetric models will be discussed. Clearly, physical real-world three-dimensional events may exhibit the required symmetry, but for larger k, the model must be closely considered since it requires a form of dependence of the random variables. The following is an explanation of this dependence. Often we assume that our data vector has independent rectangular coordinates, i.e. they may be time samples of an independent and identically distributed process or samples of independent, coincident processes. In this case, the multivariate density is the product of the marginals

$$f(\mathbf{x}) = \prod_{i=1}^{k} p(x_i)$$

The spherically symmetric model, however, requires that the multivariate density be a function of a quadratic form. It is known [16] that the only multivariate density that satisfies both of these requirements is the multivariate Gaussian.

In summary, specifying spherical symmetry for the source demands that the rectangular coordinates display a particular dependence structure. The coordinates are uncorrelated, but not independent. The exception is the multivariate Gaussian for which uncorrelatedness implies independence.

7. Conclusions

This paper presents the generalization of polar quantizers to greater than two dimensions for all spherically symmetric densities. The spherical scheme is applicable to any number of dimensions, k, and has a scalar processor implementation while optimal quantizers are available only when the k-dimensional uniform quantizer can be implemented. The derived performance expression may be used to decide if spherical schemes are of value in the particular application. In particular, for the three-dimensional generalized Gaussian source spherical quantizers always asymptotically outperform rectangular quantizers with reasonable gain in *SNR* for $c<1$. For $c \in (1,5)$ the gain is more modest.

The remarks of Section 6 demonstrate that the use of spherical coordinates quantizers requires a certain dependence structure on the rectangular coordinate realization of the source. If the source does not display the proper symmetry, the quantizers designed are no longer optimum. Similarly, optimal quantizer design also requires exact specification of the multivariate density. Rectangular quantizer design requires only knowledge of the marginal density. Their performance is constant for independent, uncorrelated or highly correlated sources as long as the marginals are known precisely. This fact suggests that rectangular coordinate quantizers are *robust* in the sense that they provide a minimum level of performance irregardless of the multivariate statistics.

Appendix A - Asymptotic Derivations

In the previous paper [5], we saw that the M_j terms were defined sequentially by ($M_0=1$)

$$M_j = \frac{\Gamma[(j+1)/2]}{\Gamma(1/2)\,\Gamma(j/2)} \int_{-\pi/2}^{\pi/2} (\sin\varphi \sin\hat{\varphi} + M_{j-1}\cos\varphi \cos\hat{\varphi}) \cos^{j-1}\varphi \, d\varphi \tag{A1}$$

When the number of levels is each quantizer, N_j, is large, the compandor model approach is appropriate. Using this model, the quantized output is approximately equal to the input plus a random error

$$\hat{\varphi} \approx \varphi + \varepsilon \frac{\Gamma(1/2)\,\Gamma[(j+2)/6]}{\Gamma[(j+5)/6] \cos^{(j-1)/3}\varphi}$$

where ε is uniformly distributed on $[-1/2N_j, 1/2N_j]$. Substituting in for $\hat{\varphi}$ in Eq.(A1), applying the trigonometric identities and the small angle approximations

$$\sin(\alpha+\beta) = \sin\alpha\cos\beta + \sin\beta\cos\alpha \quad ; \quad \sin\gamma \approx \gamma - \frac{\gamma^3}{6}$$

$$\cos(\alpha+\beta) = \sin\alpha\sin\beta + \cos\alpha\cos\beta \quad ; \quad \cos\gamma \approx 1 - \frac{\gamma^2}{2}$$

integrating over ε and simplifying yields

$$M_j \approx \frac{1 + j\,M_{j-1}}{j+1} - \frac{\pi\,\Gamma[(j+1)/2]\,\Gamma^3[(j+2)/6]}{24\,(j+5)\,\Gamma[j/2]\,\Gamma^3[(j+5)/6]} \; \frac{3 + (j+2)\,M_{j-1}}{N_j^2} \tag{A2}$$

Assume that M_j is of the form

$$M_j \approx 1 - C_j \prod_{i=1}^{j} N_i^{-2/j}$$

This holds for $j=1$ since

$$M_1 = \frac{\sin(\pi/N_1)}{(\pi/N_1)} \approx 1 - \frac{\pi^2}{6}\frac{1}{N_1^2} \rightarrow C_1 = \frac{\pi^2}{6}$$

We intend to show that this expression holds inductively. Substituting for M_{j-1} in Eq.(A2) yields

$$M_j \approx 1 - \frac{j\ C_{j-1}}{j+1}\prod_{i=1}^{j-1} N_i^{-2/(j-1)} - (j+5)\ \frac{\pi\,\Gamma[(j+1)/2]\,\Gamma^3[(j+2)/6]}{24\,\Gamma[j/2]\,\Gamma^3[(j+5)/6]} N_j^{-2}$$

$$+ \frac{\pi\,\Gamma[(j+1)/2]\,\Gamma^3[(j+2)/6]}{24\,\Gamma[j/2]\,\Gamma^3[(j+5)/6]}(j+2)\ C_{j-1}\,N_j^{-2}\prod_{i=1}^{j} N_i^{-2/(j-1)}$$

Ignoring the last term (since it is of higher power of N^{-1}) and maximizing over the value of N_j shows that the above assumption is correct and yields:

$$C_j = \frac{j^2\,C_{j-1}}{j^2-1}\ T_j$$

where

$$T_j = \left\{\frac{\pi\,\Gamma[(j+1)/2]\,\Gamma^3[(j+2)/6]}{24\,\Gamma[j/2]\,\Gamma^3[(j+5)/6]}\ \frac{(j^2-1)}{j\ C_{j-1}}\right\}^{1/j}$$

This result also yields the solution to the factorization of the number of levels in the angle quantizers:

$$N_j \approx \frac{T_j^{(j-1)/2}}{\prod_{i=j+1}^{k-1} T_i^{1/2}}\ N_{\underline{\varphi}}^{1/(k-1)}$$

A second derivative test indicates that this factorization of $N_{\underline{\varphi}}$ maximizes the value of M_{k-1} for any spherically symmetric source.

Appendix B - Density Relationship

For a spherically symmetric source, we desire a relationship between the densities of the length of the vector itself (magnitude) and the length of its projection on a rectangular coordinate axis. The length of the projection depends upon two independent events: the length of the vector and its angle from the coordinate axis under consideration. The length of the projection is the product of the radius and the length of the projection of a unit vector in the same direction.

First we consider the density of the projection of a unit vector. The rotation of a unit vector in all directions describes a unit hypersphere. The surface of this hypersphere is described by $k-1$ angle coordinates: longitude $\varphi_1 \in [0,2\pi)$ and $k-2$ latitudes, $\varphi_j \in [-\pi/2,\pi/2]$, $j=2,3,...k-1$. Consider a horizontal band on the hypersphere with $\varphi_{k-1} \in [-\sin^{-1}h, \sin^{-1}h]$ and all other angles going through their complete range. This ring corresponds to the region on the surface of the hypersphere which would project a height coordinate of x_k with $|x_k| \leq h$. We are interested in the density function of the absolute value of this height projection. Since the direction of the unit vector is random, i.e.

uniformly distributed over the surface of the hypersphere, the probability of having a projection z, $|z| \le h$, is proportional to the surface area of the ring

$$F_{|z|}(z) = \text{Prob}(|Z| \le h) \propto \int_{0}^{2\pi} \int_{-\pi/2}^{\pi/2} \cdots \int_{-\pi/2}^{\pi/2} \int_{-\sin^{-1}h}^{\sin^{-1}h} d\Sigma$$

where $d\Sigma$ is the differential surface element of the unit sphere

$$d\Sigma = r^{k-1} \cos \varphi_2 \cos^2 \varphi_3 \cdots \cos^{k-2} \varphi_{k-1} \, d\varphi_1 \, d\varphi_2 \cdots d\varphi_{k-1}$$

Evaluating the integral, the probability is proportional to

$$F_{|z|}(h) \propto \begin{cases} \sin^{-1}h + h \displaystyle\sum_{r=0}^{(k-4)/2} \frac{(r!)^2 2^{2r}}{(2r+1)!}(1-h^2)^{(2r+1)/2} & ; \ k \text{ even} \\ h \displaystyle\sum_{r=0}^{(k-3)/2} \frac{(2r)!}{(r!)^2 2^{2r}}(1-h^2)^r & ; \ k \text{ odd} \end{cases}$$

Since the length of the projection of the actual vector is the product of the magnitude r and the length z of the projection of a unit vector, the probability of the absolute value of the projection of a random vector in $\mathbf{R}^k$ being less than some value x is

$$F_{|x|}(x) = \text{Prob}(|X| \le x) = \text{Prob}(R = x/z, \ Z = z)$$

Since the projection z can only have values within $[0,1]$, this probability can be written as an integral

$$F_{|x|}(x) = \int_0^1 \text{Prob}(R{=}x/z) \, f_{|z|}(z) \, dz = \int_0^1 F_k(x/z) \, f_{|z|}(z) \, dz$$

where $f_{|z|}$ is the unit vector projection's density function. Differentiating, changing variables and noting that $f_{|x|}(x){=}2f_x(x)$ for symmetric densities yields

$$f_x(x) = \frac{1}{2} f_{|x|}(x) = \frac{1}{2}\int_0^1 f_k(x/z) \, f_{|z|}(z) \, \frac{1}{z} dz = \int_x^{\infty} f_k(y) \, f_{|z|}(x/y) \, \frac{dy}{2y}$$

For the three-dimensional case, this form can be further simplified. With $k{=}3$, the unit vector projection density reduces to $F_{|z|}(z) \propto h$ and $f_{|z|}(z)$ is uniform with value unity on $[0,1]$. The above relationship between the densities becomes

$$f_x(x) = \int_x^{\infty} f_3(y) \, \frac{dy}{2y}$$

Differentiation yields the desired result

$$f_x'(x) = -\frac{f_3(x)}{2x} \quad \rightarrow \quad -2r \, f_x'(r) = f_3(r)$$

For cases besides $k{=}3$, the form of $f_{|z|}(z)$ is not as simple and the simplifications do not result.

Acknowledgement

This research and its presentation have been supported by a Physio-Control Corporation Career Development Award and a National Science Foundation New Engineering Faculty Research Incentive Grant ECS - 8206237.

References

1. J.A. Bucklew and N.C. Gallagher Jr., "Quantization Schemes for Bivariate Gaussian Random Variables," *IEEE Trans. Inform. Theory*, Vol. IT-25, Sept. 1979, pp.537-543.
2. W.A. Pearlman, "Polar Quantization of a Complex Gaussian Random Variable," *IEEE Trans. Comm.*, Vol. COM-27, June 1979, pp.892-899.
3. J.A. Bucklew and N.C. Gallagher Jr., "Two-Dimensional Quantization of Bivariate Circularly Symmetric Densities," *IEEE Trans. Inform. Theory*, Vol. IT-25, Nov. 1979, pp.667-671.
4. S.G. Wilson, "Magnitude/Phase Quantization of Independent Gaussian Variates," *IEEE Trans. Comm.*, Vol. COM-28, Nov. 1980, pp.1924-1929.
5. P.F. Swaszek & J.B. Thomas, "k-Dimensional Polar Quantizers for Gaussian Sources," *Proc. Allerton Conf. Communication, Control and Computing*, Sept.-Oct. 1981, pp.89-97.
6. R.D. Lord, "The Use of the Hankel Transform in Statistics," *Biometrika*, Vol. 41, June 1954, pp.44-55.
7. D.K. McGraw & J.F. Wagner, "Elliptically Symmetric Distributions," *IEEE Trans. Inform. Theory*, Vol. IT-14, Jan. 1968, pp.110-120.
8. M. Kendall, *The Geometry of n Dimensions*, Griffin Co. Ltd., London, 1962.
9. W.R. Bennett, "Spectra of Quantized Signals," *Bell System Tech. Jour.*, July 1948, pp.446-472.
10. V.R. Algazi, "Useful Approximations to Optimal Quantization," *IEEE Trans. Comm. Tech.*, Vol. COM-14, June 1966, pp.297-301.
11. J. Max, "Quantizing for Minimum Distortion," *IRE Trans. Inform. Theory*, Vol. IT-6, March 1960, pp.7-12.
12. P. Zador, *Development and Evaluation of Procedures for Quantizing Multivariate Distributions*, Stanford Univ. Dissert., Stat. Dept., Dec. 1963.
13. J.H. Conway & N.J.A. Sloane, "Voronoi Regions of Lattices, Second Moments of Polytopes and Quantization," *IEEE Trans. Inform. Theory*, Vol. IT-28, March 1982, pp.211-226.
14. A. Gersho, "Asymptotically Optimal Block Quantizers," *IEEE Trans. Inform. Theory*, Vol. IT-25, July 1979, pp.373-380.
15. M. Kanefsky & J.B. Thomas, "On Polarity Detection Schemes with Non-Gaussian Inputs," *Jour. Franklin Institute*, Vol. 280, 1965, pp.120-138.
16. K. Yao, "A Representation Theorem and Its Application to Spherically-Invariant Random Processes," *IEEE Trans. Inform. Theory*, Vol. IT-19, Sept. 1973, pp.600-608.
17. R. Weinstock, *Calculus of Variations*, McGraw-Hill, N.Y., 1952.
18. H. Abramowitz and I.A. Stegun, *Handbook of Mathematical Functions*, National Bureau of Standards Applied Mathematics Series #55, Dec. 1972.

IDENTIFICATION OF CONSERVATIVE LINEAR MECHANICAL SYSTEMS[1]

S.W. SIRLIN[2]
Columbia University
New York, NY 10027

R.W. LONGMAN[3]
Columbia University
New York, NY 10027

J.N. JUANG[4]
Martin Marietta Corporation
Denver, Colorado 80201

ABSTRACT

Previous work in realization and identification theory has been concerned with general first order systems. Conservative linear mechanical systems are an important class of systems, which have a special second order form involving symmetric matrices. This paper utilizes the additional structural information available, for purposes of realization and identification. A realization is obtained from input-output data, and then knowledge of the system input and output matrices is used to determine a set of linear equations whereby we identify·the system mass and stiffness matrices. Necessary and sufficient conditions for determination of global identifiability are obtained as well as conditions which assure generic identifiability.

INTRODUCTION

Conservative linear mechanical systems have a very specialized structure generated from a Lagrangian expressable in terms of quadratic forms. On the other hand, the modern controls literature dealing with system realization and identification [1-4] treats a much more general class of systems. When doing system identification of a large system, for example in large flexible spacecraft modeling, it is important to utilize as much a priori knowledge of the system as is available, in order to minimize the amount of computation and maximize the accuracy of the result. This paper presents methods of system realization and parameter identification in conservative linear mechanical systems taking full advantage of the a priori knowledge of the structure. Identification is based on knowledge of the impulse response function and the input and output maps, extending the results of J.N. Juang and C.T. Sun [5].

The results are applicable to the general problem area of linear vibrations, but is motivated by the identification problems for future very large flexible spacecraft. The effect of damping in such structures is known to be small on the time scale of interest. It will be necessary to use control systems to not only control the attitude, but also to control the shape of such structures, sometimes to within very small tolerances dictated by the wavelength used in an antenna. The control design problem is aggravated by

1. This Research was supported by NSF Grant No. NSF-CEE-80-19275

2. Graduate Research Assistant.

3. Professor of Mechanical Engineering.

4. Member of the Technical Staff.

the fact that accurate system models cannot be generated on the ground, because tests cannot be performed on the vehicles in a one-g environment--generally these structures cannot support their own weight under gravity. Hence, a rough control system must be designed and used until on orbit tests for identification purposes can be performed. Based on these tests, the attitude and shape controller would be tuned to obtain the desired performance. Also, certain applications will require construction of the structure in space, and simultaneously controlling the structure. These applications will require periodic identification of the structural dynamics as the structure develops. Certain tests can be performed on the ground, for example it is likely that accurate information about mass and inertia properties would be obtainable. This, together with the vibratory nature of the system, generates an identification problem with a special mathematical structure, and knowledge of this structure is incorporated into the identification methods developed here.

SYSTEM STRUCTURE OF CONSERVATIVE MECHANICAL SYSTEMS

The class of conservative linear mechanical systems may be described by the following set of equations

$$\begin{aligned} &M\ddot{q} + Kq = Bu \\ &y = Cx;\ X^T = [q^T \dot{q}^T] \qquad C = [Cp\ Cv] \end{aligned} \tag{1}$$

where M is symmetric positive definite, K is symmetric positive semidefinite, q is a position vector of length n, u is an input vector of length m, and y is an output vector of length p. All quantities are real. Due to the nature of M and K, we may transform to modal form (see [6])

$$\begin{aligned} &\ddot{\eta} + \Lambda\eta = \hat{B}u \\ &y = \hat{C}\xi,\ \hat{C} = [CpP\ CvP],\ \xi^T = [\eta^T\ \dot{\eta}^T] \\ &q = P\eta,\ P^TMP = I,\ P^TKP = \Lambda,\ \hat{B} = P^TB \end{aligned} \tag{2}$$

where I is the identity matrix and Λ is diagonal. We may write either of the above in first order form,

$$\begin{aligned} &\dot{x} = Ax + Gu \\ &y = Cx \end{aligned} \tag{3}$$

where $A = \begin{bmatrix} 0 & I \\ M^{-1}K & 0 \end{bmatrix}$, $G = \begin{bmatrix} 0 \\ M^{-1}B \end{bmatrix}$;

or $\dot{\xi} = \hat{A}\xi + \hat{G}u$

$$\begin{aligned} &y = \hat{C}\xi \\ &\hat{A} = \begin{bmatrix} 0 & I \\ -\Lambda & 0 \end{bmatrix},\ \hat{G} = \begin{bmatrix} 0 \\ \hat{B} \end{bmatrix} \end{aligned} \tag{4}$$

We would like to determine M and K uniquely from a combination of impulse response data and knowledge of C and B. We do this by first finding a realization of the form (4), and then determining the matrix P.

REALIZATION FROM IMPULSE RESPONSE DATA

Given a system of the form

$$
\begin{aligned}
\dot{z} &= \bar{A}z + \bar{G}u \\
y &= \bar{C}z
\end{aligned}
\tag{5}
$$

where $\bar{A} = \begin{bmatrix} 0 & I \\ F & 0 \end{bmatrix}$, $\bar{G} = \begin{bmatrix} 0 \\ \bar{B} \end{bmatrix}$, $\bar{C} = [\bar{C}p\ \bar{C}v]$

$(\bar{C},\bar{A},\bar{G})$ is complete, and the system dimensions are the same as in (3), note that

$$
\begin{aligned}
\bar{A}^k &= \begin{bmatrix} F^{\frac{k}{2}} & 0 \\ 0 & F^{\frac{k}{2}} \end{bmatrix}, \text{ k even} \\
&= \begin{bmatrix} 0 & F^{\frac{k-1}{2}} \\ F^{\frac{k+1}{2}} & 0 \end{bmatrix}, \text{ k odd,}
\end{aligned}
\tag{6}
$$

so that impulse response Taylor series coefficients are

$$
\begin{aligned}
Y_k = \bar{C}\,\bar{A}^k\,\bar{G} &= \bar{C}v\,F^{\frac{k}{2}}\,\bar{B}, \text{ k even} \\
&= \bar{C}p\,F^{\frac{k-1}{2}}\,\bar{B}, \text{ k odd}
\end{aligned}
\tag{7}
$$

If we expect the pair $(Cp, -M^{-1}K)$ to be observable (this is the case if $Cv = 0$), then we may use the Y_k, k odd, to obtain, by Ho's algorithm, (presented in [3]) matrices $\bar{C}p$, F, $\bar{B}$. We may then obtain $\bar{C}v$ from

$$
\begin{aligned}
[Y_0\ Y_2 \cdot\cdot\cdot\ Y_{2n-2}] &= \bar{C}v[\bar{B}\ F\bar{B} \ldots F^{n-1}\bar{B}] \\
&= \bar{C}vPc
\end{aligned}
\tag{8}
$$

The matrix Pc, which is nxnm, must be of rank n if the original system is controllable. Note that if we use the above procedure when $(Cp, -M^{-1}K)$ is not observable, Ho's algorithm will yield a system of order $< n$, implying the need for a more complex procedure.

Even if $(Cp, -M^{-1}K)$ is not observable, we may use the knowledge of A, G to obtain a realization more easily than for a general 2nx2n system. Set

$$
\bar{D} = \begin{bmatrix} \bar{C}v \\ \bar{C}p \end{bmatrix}
\tag{9}
$$

and consider the observability matrix for the system (C,A)

$$
Po = \begin{bmatrix}
\bar{C}p & \bar{C}v \\
\bar{C}vF & \bar{C}p \\
\cdot & \cdot \\
\cdot & \cdot \\
\vdots & \vdots \\
\bar{C}pF^{n-1} & \bar{C}vF^{n-1} \\
\bar{C}vF^{n} & \bar{C}pF^{n-1}
\end{bmatrix}
\tag{10}
$$

Clearly, if there exists a $y \in R^n$,

$$
y \in \bigcap_{i=0}^{n-1} \ker\,(\bar{D}F^i),
\tag{11}
$$

then $x \in \ker\,(Po)$, where $x^T = [0\ y^T]$. Therefore, if $(\bar{C},\bar{A})$ is observable, then so is $(\bar{D},F)$. We may thus use Ho's algorithm on the Markov parameters for this new system,

$$
Z_k = \begin{bmatrix} Y_{2k} \\ Y_{2k+1} \end{bmatrix}
\tag{12}
$$

to obtain a realization $(\bar{D},F,\bar{B})$. In the scaler case this leads to a generalized Hankel matrix of size 2nxn, rather than one of size 2nx2n. In general, the Hankel matrix is reduced by two, or by a factor tending to two as the system order grows.

IDENTIFICATION OF M AND K FROM THE REALIZATION

Assume that we now have a realization $(\bar{C},\bar{A},\bar{G})$ of the form (5) obtained from impulse response data of system (1). Defining

$$D = \begin{bmatrix} Cv \\ Cp \end{bmatrix}, \quad \hat{D} = \begin{bmatrix} \hat{C}v \\ \hat{C}p \end{bmatrix} \tag{13}$$

we see that the two systems $(D, -M^{-1}K, M^{-1}B)$ and $(\bar{D},F,\bar{B})$ have the same impulse response and are both of minimal order. The two must therefore be related by a similarity transformation [7, p. 499]. The matrix F is therefore diagonalizable, and we may obtain an equivalent realization $(\tilde{D},\Lambda,\tilde{B})$, from the system $(\bar{D},F,\bar{B})$. This system is similar to the system $(\hat{D},\Lambda,\hat{B})$, and so a nonsingular matrix Γ exists such that

$$\begin{aligned} \eta(t) &= \Gamma\tilde{\eta}(t) \\ \hat{\Lambda}\Gamma &= \Gamma\Lambda \\ \tilde{D} &= D\,P\,\Gamma \end{aligned} \tag{14}$$

$$\tilde{B} = \Gamma^{-1}P^{T}B \tag{15}$$

We now have a linear relation between our known input and output matrices B,D and those derived from input output data, $\tilde{B},\tilde{D}$. If these equations can be solved for P, we may obtain M and K from

$$\begin{aligned} M &= (P^{-1})^{T}P^{-1} \\ K &= (P^{-1})^{T}\Lambda\,P^{-1} \end{aligned} \tag{16}$$

It is also true that

$$\tilde{D}\,\tilde{B} = D\,M^{-1}B$$

$$\tilde{D}\,\Lambda\,\tilde{B} = D(M^{-1}K\,M^{-1})B \tag{17}$$

and therefore if both B and D are of rank n, we may easily obtain M and K. This, however, requires $2p \geqslant n$ and $m \geqslant n$, a large number of sensors and actuators. We now recast equations (14) and (15) in order to determine when a lesser number of sensors and actuators is sufficient.

EQUATIONS FOR DETERMINING THE TRANSFORMATION MATRICES

We should like to determine the modal matrix P, however it is not unique, and hence the matrix Γ is not unique as well. The amount of uncertainty is limited, however. For example, if the eigenvalues of $M^{-1}K$ are distinct, we may only replace an arbitrary set of modal vectors (columns of P) by the negative of those vectors. Further, although the matrix Γ is not unique, its square $\Gamma^{T}\Gamma$ is. This is demonstrated by the following.

Lemma 1. *If two nonsingular matrices* P, $\bar{P}$ *both diagonalize matrices* M, K *as in (2), then there exists a nonsingular matrix* V *such that* $\bar{P} = PV$, *and*

$$\begin{aligned} VV^{T} &= I \\ \Lambda V &= V\Lambda \end{aligned} \tag{18}$$

Proof. Clearly a nonsingular transformation V relating $P,\bar{P}$ exists, and then

$$I = \bar{P}^T M \bar{P} = V^T P^T M P V = V^T V$$

while

$$\Lambda = \bar{P}^T K \bar{P} = V^T \Lambda V,$$

so V satisfies (18).

The map $P\Gamma$: $\tilde{\eta}(t) \rightarrow q(t)$ is unique, due to controllability, hence given a $\bar{P}$ as in the above lemma, we can find a $\bar{\Gamma}$ (which must satisfy (14) as Γ does) such that $P\Gamma = \bar{P}\bar{\Gamma}$, or $\Gamma = V\bar{\Gamma}$. Therefore $\Gamma^T\Gamma = \bar{\Gamma}^T V^T V \bar{\Gamma} = \bar{\Gamma}^T\bar{\Gamma} \triangleq \Phi$ is unique. Note that, as Λ is symmetric, Γ^T also satisfies (14), and hence so does Φ. Clearly the matrix Φ is symmetric and positive definite, so it possesses a unique symmetric, positive definite square root, which will now be shown to satisfy (14).

Lemma 2. Given a diagonal matrix $S \geqslant 0$, which satisfies

$$AS^2 = S^2 A \tag{19}$$

then

$$AS = SA \tag{20}$$

Proof. In indicial notation, (19) becomes

$$A_{ij}(S_j^2 - S_i^2) = 0 \qquad \forall\, i,j,$$

which implies

$$A_{ij}(|S_j| - |S_i|) = 0$$

which yields (20), as $S_i \geqslant 0$ for all i.

Lemma 3. Given a symmetric $S \geqslant 0$, satisfying (19), also satisfies (20).

Proof. We can always write $S = Q^T \Delta Q$, where Q is orthogonal and $\Delta \geqslant 0$ is diagonal. Then (19) yields

$$(Q\,A\,Q^T)\Delta^2 = \Delta^2(Q\,A\,Q^T).$$

Using Lemma 2, we get

$$(Q\,A\,Q^T)\Delta = \Delta(Q\,A\,Q^T)$$

which yields (20).

Our goal is to first determine Φ, then choose a Γ satisfying (14) and $\Gamma^T\Gamma = \Phi$, and finally to determine a modal matrix P. While many choices of Γ are possible, Lemma 3 shows that the positive definite symmetric square root of Φ, which is unique, is among them.

Examining (15), it is evident that if we set

$$R = \begin{bmatrix} D \\ B^T \end{bmatrix}, \; \tilde{R} = \begin{bmatrix} \tilde{D}\,\Gamma^{-1} \\ \tilde{B}^T\,\Gamma^T \end{bmatrix} \tag{21}$$

so that (15) becomes

$$R\,P = \tilde{R}, \tag{22}$$

and if rank R = n, which requires $2p+m \geq n$, then we may determine the matrix P as a unique function of Γ by

$$P = R^{\dagger}\tilde{R} = (R^T R)^{-1} R^T \tilde{R} \tag{23}$$

Note that even if Γ is known, we cannot determine P uniquely unless rank R = n, hence this is necessary for identifiability of M and K. If all the sensors are colocated with the actuators, then rank R = rank B^T, and we will require many more sensors and actuators to identify M and K than if we had not colocated. Hence colocation, while it may be useful from a control standpoint, is not useful for identification.

Determination of Φ. We now need to determine Γ (or Φ). If rank R = n, but R is square, then any Γ will satisfy (15) if P is chosen as in (22). In order to determine Γ, therefore, R cannot be square, and we will require $2p+m > n$. Substituting (23) into (22), we obtain

$$(R\,R^{\dagger} - I)\,\tilde{R} = 0 \tag{24}$$

or

$$\bar{X}\,\Phi = \bar{Y}, \tag{25}$$

where

$$\bar{X} = \hat{X}\,\tilde{B}^T = X\,P\Gamma^{-T}$$

$$X = \hat{X}\,B^T$$

$$\bar{Y} = \hat{Y}\,\tilde{D}$$

$$R\,R^{\dagger} - I = [\hat{Y}, -\hat{X}]$$

$$R\,R^{\dagger} = \begin{bmatrix} D\,\check{R}\,D^T & D\,\check{R}\,B \\ B^T\check{R}\,D^T & B^T\check{R}\,B \end{bmatrix}$$

$$\check{R} = (R^T R)^{-1}$$

Equations (25), together with (14) and the fact that Φ is symmetric positive definite, are all we have to determine Φ. Note that both the right hand side and the left hand side of (25) contain terms which are not known before a realization is obtained, $(\tilde{B}, \tilde{D})$ and hence, in general, the identifiability of the system is not known a priori.

We next determine what class of matrices satisfy (14). Suppose Λ has $r \leq n$ distinct eigenvalues λ_i, each of multiplicity μ_i. Then we assume Λ to be arranged as

$$\Lambda = \begin{bmatrix} \Lambda_1 & 0 & \cdots & 0 \\ 0 & & & \vdots \\ \vdots & & & 0 \\ 0 & \cdots & 0 & \Lambda_r \end{bmatrix} \tag{26}$$

$$\Lambda_i = \lambda_i I, \quad \mu_i \times \mu_i$$

It is easy to show that $\Lambda\Phi = \Phi\Lambda$ if and only if the only nonzero terms in Φ are the $\mu_i \times \mu_i$ blocks Φ_i along the diagonal. We then have r separate problems to solve for each block Φ_i, $i \in \underline{r}$ (note that $\underline{r} = \{1, 2, \ldots, r\}$)

$$\begin{gathered} \bar{X}_i\,\Phi_i = \bar{Y}_i \\ \Phi_i^T = \Phi_i, \quad \Phi_i > 0 \end{gathered} \tag{27}$$

where we have partitioned $\bar{X} = [\bar{X}_1, . . . X_r]$, and $\bar{Y}$ similarly.

Theorem 1. (Symmetric solution of equations.) The consistent system of equations

$$\begin{aligned} &(a)\ A\ X = B \\ &(b)\ X^T = X \\ &(c)\ X > 0 \end{aligned} \tag{28}$$

where A, B are $n \times m$ matrices, has a unique solution X, $m \times m$, if and only if rank $A = m$.

Proof. To establish the necessity, consider a general singular value decomposition [8] of A where rank $A = s < m$,

$$A = U \Sigma V^T = U_1\ S\ V_1^{\ T}$$

where

$$U = [U_1\ U_2],\ V = [V_1\ V_2],\ \Sigma = \begin{bmatrix} S & 0 \\ 0 & 0 \end{bmatrix}.$$

Any solution of (a) may then be written as

$$X = A^{\dagger} B + V_2 \beta \tag{29}$$

where $A^{\dagger} = V\, \Sigma^{\dagger} U^T = V_1 S^{-1} U_1^{\ T}$, and β is an arbitrary $(m-s) \times m$ matrix. Setting

$$Z = \begin{bmatrix} Z_1 & Z_2 \\ Z_3 & Z_4 \end{bmatrix} = U^T B V$$

$$\beta = \gamma V^T,\ \gamma = [\delta, \alpha],$$

where Z_1 is $s \times s$, and δ is $(m-s) \times s$, we may then write (29) as

$$X = V \begin{bmatrix} S^{-1} Z_1 & S^{-1} Z_2 \\ \delta & \alpha \end{bmatrix} V^T \tag{30}$$

As V is orthonormal, to satisfy (b) we need

$$S^{-1} Z_1 = Z_1^{\ T} S^{-1} \tag{31}$$

$$\delta = (S^{-1} Z_2)^T$$

$$\alpha = \alpha^T \tag{32}$$

Clearly (31) is a constraint on B so that a symmetric solution of (28a) exists, and so is an additional consistency constraint, in addition to those required for (a) alone to have a solution. Equations (32) serve to determine β in (29). Note that the $(m-s) \times (m-s)$ matrix α is required only to be symmetric, so that a solution to (a,b) is never unique if $m-s > 0$. The addition of (28c) will produce m inequality constraints on the elements of α. As the inequalities are strict, the set of solutions α (assuming at lease one solution -a further consistency requirement) is an open set, and therefore (a,b,c) has no unique solution if $m > s$.

To establish sufficiency, suppose rank $A = m$, then the solution to (28a) is unique, hence at most one solution to (28) exists. If the equations are consistent, this must be a solution.

The point of the theorem is that our knowledge that X is symmetric and positive definite does not help us determine it uniquely. Applying the Theorem to (27), we may solve for Φ uniquely if and only if

$$\text{rank } \bar{X}_i = \mu_i \text{ , } i \in \underline{r}. \tag{33}$$

The matrices $\bar{X}_i$ are not known a priori, however. Noting that Γ must have the same structure as $\bar{\Phi}$, and partitioning P as $\bar{X}$, we may write

$$\bar{X}_i = X\, P_i\, \Gamma_i^{-T} \tag{34}$$

and as the Γ_i are square and non-singular,

$$\text{rank } \bar{X}_i = \text{rank } X\, P_i . \tag{35}$$

The matrix X is known a priori, but P is not. As the columns of P are linearly independent, P_i is of full rank so that $\bar{X}_i$ is of full rank if and only if

$$0 = (\text{Image } P_i) \cap (\text{kernal } X). \tag{36}$$

We can now state a priori conditions for identifiability of a system.

IDENTIFIABILITY CONDITIONS

Theorem 2. *The matrices M, K are identifiable if and only if*

a) rank R = n, and
b) Φ is uniquely determined.

Proof. Clearly if Φ is available and rank R = n, we may set $\Gamma = \Phi^{\frac{1}{2}}$, then solve (23) for P, and obtain M, K from (16).

To show necessity, first suppose a) is not true, so rank R<n. Then even if we determine a Γ, we do not obtain a unique P, for if we have a nonsingular solution to (22) P , then any matrix $\bar{P} = P + k\Delta$, where $\Delta \in \ker R$, k is real, is also a solution to (22). The requirement that $\bar{P}$ be nonsingular can be met by all but a finite number of real k. Now if P , $\bar{P}$ are both to determine the same matrices M,K by (16), then they must satisfy Lemma 1. Writing $\bar{P} = P\,V$ we see that $V = I + P^{-1}\Delta k$, and the first requirement of Lemma 1 becomes

$$0 = V^TV - I = k(k\Delta^T P^{-T} P^{-1}\Delta + \Delta^T P^{-T} + P^{-1}\Delta)$$

which allows at most two solutions for k. Therefore any two solutions P , $\bar{P}$ to (22), with known Γ, will not, in general, produce the same system matrices M,K through (16).

Next suppose a) is satisfied but b) is not, i.e. rank $\bar{X}_i < \mu_i$ for some $i \in \underline{r}$. Then given any symmetric, positive definite solution Φ , we have an infinite number of other solutions $\bar{\Phi} = \Phi + k\Delta$, where $\Delta \in \ker \bar{X}$, $\Delta^T = \Delta$, Δ commutes with Λ, and k is real. The requirement that k be positive definite yields a set of strict inequalities on k, or $k \in \Upsilon$. The set Υ is open, and hence we still have an infinite number of possible k. Finding appropriate $\bar{\Gamma}$, Γ , we can write $\bar{\Gamma} = \Gamma + \delta(k)$, where

$$\Gamma^T\delta + \delta^T\Gamma + \delta^T\delta = k\Delta.$$

Now for Φ , $\bar{\Phi}$ to yield the same M,K, using Lemma 1 we must have $\bar{\Gamma} = V\Gamma$; where V satisfies (18). Now $V = I + \delta\Gamma^{-1}$, so

$$0 = V^TV-I = \Gamma^{-T}\delta^T + \delta\Gamma^{-1} + \Gamma^{-T}\delta^T\delta\Gamma^{-1}$$

or

$$0 = k\Delta.$$

Therefore Φ , $\bar{\Phi}$ lead to different system matrices unless $\Phi = \bar{\Phi}$, and so we cannot identify M,K uniquely if we cannot determine Φ uniquely.

The main result follows.

Theorem 3. An arbitrary system of the form (1) is identifiable, in the sense that the matrices M and K can be uniquely determined from knowledge of input and output maps B,C, and input-output data,

a) for all possible matrices M and K, if and only if rank B = rank D=n

b) for almost all matrices M and K (i.e. generically) if rank R = n and rank X ≥ 1*, where R is defined in (21) and X in (25). These conditions can be met if 2p+m* $\geq$ *n.*

Proof. a) The sufficiency of these is apparent from (17). It is apparent from Theorem 1 that for a system to be identifiable it is necessary that (36) be true for all $i \in \underline{r}$. This is hardest to satisfy when all of the eigenvectors are repeated, so r = 1. In that case (36) becomes

$$0 = \ker X$$

or

$$\text{rank } X = n$$

From the definition of $\bar{X}$, $\bar{Y}$ in (25), this clearly requires that $\hat{X}$, B, $\bar{Y}$, $\hat{Y}$, $\tilde{D}$, D all be of rank n.

b) By Theorem 2, we clearly need rank $\underline{R}$ = n. Suppose rank X = s. Using singular value decomposition, $X = U\Sigma V^T$. As V is orthonormal we can write

$$P_i = V\bar{P}_i = V\begin{bmatrix}\hat{P}_i \\ \hat{Q}_i\end{bmatrix}, \quad i \in \underline{r}$$

where $\hat{P}_i$ is $s \times \mu_i$, $\hat{Q}_i$ is $(n-s) \times \mu_i$. Then (33,35) become

$$\begin{aligned} \mu_i &= \text{rank } (XP_i) = \text{rank } (U\Sigma\bar{P}_i) \\ &= \text{rank } (\Sigma\bar{P}_i) \\ &= \text{rank } \hat{P}_i, \ i \in \underline{r}. \end{aligned} \tag{37}$$

Generically $\hat{P}_i$ will be of full rank, so (37) is equivalent to rank $X \geq \mu_i, \forall i \in \underline{r}$. Furthermore, generically, the system eigenvalues will all be distinct so that $\max \mu_i = 1$.

Similar techniques to those used above may be applied in order to identify first order systems with symmetric system matrices.

SEPARATE POSITION AND VELOCITY SENSORS

A useful special case is one in which each output contains either position or velocity information, but not both. In this case we may write the system output matrix as

$$C = \begin{bmatrix} C_p & 0 \\ 0 & C_v \end{bmatrix} \quad p \times 2n$$

From (7) it is evident that many of the terms in the generalized Hankel matrix are zero, so that more simplification is possible. We can again define $\bar{D}$, Z, D, $\hat{D}$ as in (9,12,13), but these are now of smaller dimension. The previous results hold specifically Theorem 3 provided we use p in place of 2p.

CONCLUSION

We have shown under what conditions the mass and stiffness matrices of a conservative linear mechanical system may be uniquely identified, and how this may be done. Future work will consider the extension of these results to systems with known mass matrix, gyroscopic or damped second order systems, as well as to discrete rather than continuous measurements.

REFERENCES

1. R. Bellman and K.J. Åström, "On Structural Identifiability," Mathematical Biosciences, Vol. 7, 1970, pp. 329-339.

2. K. Glover and J.C. Willems "Parametrizations of Linear Dynamical Systems: Canonical Forms and Identifiability," IEEE Transactions on Automatic Controls, Vol. AC-19, 1974,pp. 640-645.

3. B.L. Ho and R.E. Kalman, "Models from Input/Output Data," Proceedings of the 3rd Annual Allerton Conference on Circuit and System Theory, 1965.

4. A.J. Tether, "Construction of Minimal Linear State-Variable Models from Finite Input-Output Data," IEEE Transactions on Automatic Controls, Vol. AC-15, 1970, pp. 427-436.

5. J.N. Juang and C.T. Sun, "System Identification of Large Flexible Structures Using Simple Continuum Models," The Journal of the Astronautical Sciences, to appear.

6. L. Meirovith, Analytical Methods in Vibrations, Macmillan, 1967.

7. T.E. Fortmann and K.L. Hitz, An Introduction to Linear Control Systems, Marcel Dekker, 1977.

8. A. Ben Israel and T.N.E. Greville, Generalized Inverses, R.E. Kreuger, 1980.

A METHOD FOR SIMULTANEOUS ESTIMATION OF NOISE COVARIANCES AND PLANT IDENTIFICATION

Y.G. ZHANG[1]
Institute of System Science
Academica Sinica
Beijing, P.R.C.

R.W. LONGMAN[2]
Columbia University
New York, NY 10027

ABSTRACT

Discrete linear dynamic systems with plant and observation noise are considered. A method is presented for simultaneous estimation of the measurement noise covariance, and the plant noise covariance, by solution of a set of linear equations. These equations are expressed in a particularly simple explicit form by making use of Kronecker product notation. In addition, a method is given for identification of the plant dynamics from the observation correlation. This method also involves only solution of linear equations, and it can be computed recursively along with the noise covariance estimation. After establishing a martingale convergence result, it is shown that the estimates of these matrices converge (a.s.) to the true values as the number of samples increases.

INTRODUCTION

In physical systems it is often impossible or impractical to make specific experiments to isolate and determine system parameters and to determine noise characteristics. Hence, any simple method of using an output record to estimate the plant and observation noise covariances and the system matrix (or a subset of these) has many applications, for example in adaptive filtering. This paper presents such a method.

Previous related work by other authors include references [1,2] of Mehra which develop estimation methods for plant and observation noise covariances. Reference [1] uses correlations of the innovations to develop its estimates, and is limited to estimation of mn unknowns in the plant noise covariance, where m is the dimension of the observations, and n that of the plant. Reference [2] offers other approaches to obtaining the noise covariances. In reference [3] as modified in [4] the correlation of the difference between true filter and one step predictor estimates is used to obtain the noise covariances. In the present work, we obtain simultaneous estimates of both plant and observation noise and prove they satisfy the strong consistency property. By comparison to [1], there is no limitation on the number of elements of the plant noise to be estimated. The method has an efficiency advantage over the other approaches, when the aim is purely to obtain these covariances, since there is no need to generate estimates of the Kalman gain.

Various methods for system identification exist in the literature [5,6], although no method can handle the fully general multiple variable case. Some methods require iterative solution of least squares equations. Others shift to time series form and identify input-output relations in the

[1] Presently, Visiting Scholar, Department of Mathematical Statistics, Columbia University, New York, NY 10027.

[2] Professor of Mechanical Engineering.

z transform domain without determining the system matrix itself. The method presented here is applicable to certain classes of multivariable systems, and uses an observation correlation approach. It is particularly simple in that it results in a linear system of equations to solve, which can be given in recursive form. It can be calculated in conjunction with a recursive estimation of the noise covariances, and the strong consistency property is shown to be satisfied.

PROBLEM STATEMENT

We consider a discrete time-invariant linear system

$$x(k+1) = A\,x(k) + \Gamma w(k) \tag{1}$$

$$z(k) = C\,x(k) + v(k) \tag{2}$$

where $x(k)$ is the n-dimensional state, $z(k)$ the m-dimensional observation vector ($m \leq n$), $w(k)$ a p-dimensional plant noise, and $v(k)$ an m-dimensional observation noise. It is assumed that both C and Γ are known, and the eigenvalues of $A \otimes A$ lie inside the unit circle.

The statistical properties of the noises are: $E\,w(k) = 0$, $E\,v(k) = 0$, $E\,w(k)\,v^T(j) = 0$ for all k,j. $E\,w(k)\,w^T(j) = 0$ and $E\,v(k)\,v^T(j) = 0$ for $k \neq j$, $E\,X(0)\,w^T(k) = 0$, $E\,X(0)\,v^T(k) = 0$, and $E\,x(0)\,x^T(0) > 0$. The covariances will be denoted by $\operatorname{cov} w(k) = Q$, $\operatorname{cov} v(k) = R$.

It is assumed that the pair (A,Γ) is controllable and that the system is observable, i.e. rank $[C^T, A^T C^T, \ldots, (A^{n-1})^T C^T] = n$. Hence, we can assume there is an integer q, $n/m \leq q \leq n$, such that rank of $[C^T, A^T C^T, \ldots, (A^{q-1})^T C^T] = n$. One can always use $q = n$ when the minimum q is unknown.

The matrices A,Q and R are considered unknown, and are to be identified.

ESTIMATION OF PLANT AND OBSERVATION NOISE COVARIANCES

The Basic Equations

Combining equations (1) and (2) one obtains

$$z(k+1) = CA\,X(k) + C\Gamma w(k) + v(k+1)$$

$$z(k+\ell) = CA^{\ell}\,X(k) + \sum_{j=0}^{\ell-1} CA^{j}\Gamma w(k+\ell-j-1) + v(k+\ell) \tag{3}$$

$$\ell = 1, 2, 3, \ldots$$

Let

$$Z(k) = [z^T(k), z^T(k+1), \ldots, z^T(k+q)]^T$$

$$V(k) = [v^T(k), v^T(k+1), \ldots, v^T(k+q)]^T \tag{4}$$

$$U(k) = [u_0^T(k), u_1^T(k), \ldots, u_q^T(k)]^T$$

$$\Omega_q = [C^T, (CA)^T, \ldots, (CA^q)^T]^T$$

and Ω_q^- be a generalized inverse of Ω_q such that $\Omega_q^-\Omega_q = I_n$, where

$$u_i(k) = \begin{cases} 0 & \text{if } i = 0 \\ \sum_{j=0}^{i-1} CA\,\Gamma w(k+i-j-1) & \text{if } i = 1, 2, \ldots, q \end{cases} \tag{5}$$

In terms of these new variables, one can obtain an aggregate observation equation for a q-step observation history as

$$Z(k) = \Omega q\ X(k) + V^*(k) \qquad k = 1,\ 2,\ .\ .\ . \tag{6}$$

where $V^*(k) = U(k) + V(k)$.

In terms of the following definitions

$$\mathfrak{B} = \begin{bmatrix} C & & & 0 \\ CA & \ddots & & \\ \vdots & & \ddots & \\ CA^{q-1} & \cdots & CA & C \end{bmatrix}, \qquad W(k) = \begin{bmatrix} w(k) \\ w(k+1) \\ \cdots \\ w(k+q-1) \end{bmatrix} \tag{7}$$

and using the definition of the Kronecker product $A \times B$ of any two matrices A,B [7], one can write

$$U(k) = \left[\begin{array}{c} 0m\times 1 \\ \hline \mathfrak{B}(Iq\otimes\Gamma)\ W(k) \end{array}\right]$$

$$E\ U(k)\ U^T(k) = \left[\begin{array}{c:c} 0m\times m & 0m\times mq \\ \hline 0mq\times m & \mathfrak{B}(Iq\otimes\Gamma)(Iq\otimes Q)(Iq\otimes\Gamma)^T\mathfrak{B}^T \end{array}\right] \tag{8}$$

$$E\ V(k)\ V^T(k) = I_{q+1}\otimes R$$

Then the covariance of the V* of equations (6) and (7) becomes

$$R_{v*} \quad E\ V^*(k)V^{*T}(k)$$

$$= \left[\begin{array}{c:c} Rv^*(1,1) & 0m\times mq \\ \hline 0mq\times m & Rv^*(2,2) \end{array}\right] \tag{9}$$

where $R_{v*}(1,1)$ is m×m, $R_{v*}(2,2)$ is mq×mq, and they are given by

$$R_{v*}(1,1) = R \tag{10}$$

$$R_{v*}(2,2) = \mathfrak{B}(Iq\otimes\Gamma)(Iq\otimes Q)(Iq\otimes\Gamma)^T\mathfrak{B}^T + (Iq\otimes R) \tag{11}$$

The second equation can be rewritten

$$\mathrm{vec}\ [R_{v*}(2,2) - (Iq\otimes R)]$$

$$= (\mathfrak{B}\otimes\mathfrak{B})\ \mathrm{vec}\ (Iq\otimes\Gamma Q\Gamma^T)$$

$$= \left(\begin{bmatrix} C \\ CA \\ \\ CA^{q-1} \end{bmatrix} \otimes \begin{bmatrix} C \\ CA \\ \\ CA^{q-1} \end{bmatrix} + \begin{bmatrix} 0 \\ C \\ \\ CA^{q-2} \end{bmatrix} \otimes \begin{bmatrix} 0 \\ C \\ \\ CA^{q-2} \end{bmatrix} + \ldots + \begin{bmatrix} 0 \\ \\ 0 \\ C \end{bmatrix} \otimes \begin{bmatrix} 0 \\ \\ 0 \\ C \end{bmatrix} \right) \mathrm{vec}\ \Gamma Q\Gamma^T$$

$$\triangleq \mathfrak{B}^*\ (\Gamma\otimes\Gamma)\ \mathrm{vec}\ Q \tag{12}$$

where vec Q is the vector of columns of Q in succession [7]. Equation (12) is a set of linear equations in the elements of Q. Making use of the observability assumption, if $p \leq n$, and $R^*(2,2)$, R are known, then one can solve equation (11) for vec Q.

Equations (10) and (12) constitute the desired basic equations.

Estimation of R_{v*}

In order to obtain an estimate $\widehat{R}_{v*}$ of R_{v*}, define

$$P(\ell,k) = E\ Z\ (k+\ell)\ Z^T(k) \qquad \ell = 0,\ 1,\ 2,\ .\ .\ .$$

$$Q_x(k) = E\,X(k)\,X^T(k) \tag{13}$$

From equation (6) and the definition of v*(k)

$$P(o,k) = \Omega_q\,Q_x(k)\Omega_q^T + R_{v*}$$

$$P(q+1,k) = \Omega_q\,A^{q+1}Q_x(k)\Omega_q^T \tag{14}$$

Since $Q_x(k)$ is convergent as $k\to\infty$, both $P(o,k)\to P(o)$ and $P(q,k)\to P(q)$ as $k\to\infty$, with

$$P(o) = \Omega_q\,Q_x\Omega_q^T + R_{v*}$$

$$P(q+1) = \Omega_q\,A^{q+1}Q_x\Omega_q^T \tag{15}$$

Note that z(k) asymptotic stationary ergodic under the assumption that all eigenvalues of $A\otimes A$ lie inside the unit circle, and that we can use the sample moment in place of the expectation in $P(\ell,k)$, i.e.

$$\hat{P}(N,0) = \frac{1}{N}\sum_{k=1}^{N} Z(k)\,Z^T(k)$$

$$\hat{P}(N,q+1) = \frac{1}{N}\sum_{k=1}^{N} Z(k+q+1)\,Z^T(k) \tag{16}$$

Making use of equation (15) we obtain the estimates

$$\widehat{Q_x\Omega_q^T}(N) = \left(A^{q+1}\right)^{-1}\bar{\Omega}_q\,\hat{P}(N,q+1)$$

$$\hat{R}_{v*}(N) = \hat{P}(N,0) - \Omega_q\,\widehat{Q_x\Omega_q^T}(N) \tag{17}$$

or, if A is unknown and being estimated, then

$$\widehat{Q_x\Omega_q^T}(N) = \left(\hat{A}^{q+1}\right)^{-1}\hat{\bar{\Omega}}_q\,\hat{P}(N,q+1)$$

$$\hat{R}_{v*}(N) = \hat{P}(N,0) - \hat{\Omega}_q\,\widehat{Q_x\Omega_q^T}(N) \tag{18}$$

The Noise Covariance Estimates

The desired estimates of the noise covariances can be written by defining Rv*(1,1) and Rv*(2,2) as the m×m and mq×mq partitions of $\hat{R}v*(N)$ analogous to those of Rv* in (9), and using

$$\hat{R}(N) = \hat{R}^N_{v*}(1,1)$$

$$\mathrm{vec}\,[\hat{R}^N_{v*}(2,2) - (I_q\otimes\hat{R}(N))] = \mathcal{B}^*(\Gamma\times\Gamma)\,\mathrm{vec}\,\hat{Q}(N) \tag{19}$$

to find $\hat{R}(N)$ and $\hat{Q}(N)$. If A is also being estimated, the right hand side of the latter expression is $\hat{\mathcal{B}}^*(\Gamma\otimes\Gamma)\,\mathrm{vec}\,Q$, where $\hat{\mathcal{B}}^*$ indicates use of $\hat{A}$ for A in $\mathcal{B}^*$.

Alternative Methods for Special Cases

Analogous techniques can be used in two special situations. Suppose first that we are interested only in estimating R. Then a simple approach applies as described by Mehra [2]. Consider

$$Q_z(i,k) = E[z(k+i)\,z^T(k)] \tag{20}$$

then

$$Q_z(o,k) = C\,Q(k)C^T + R$$

$$Q_z(\ell,k) = CA^{\ell} Q_x(k)C^T \qquad = 1, 2, 3, ... \tag{21}$$

Since $Q_x(k)$ is convergent under the assumption that all eigenvalues of A×A lie inside the unit circle, the $Q_z(\ell,k)$, $\ell = 0, 1, 2, . . .$ are as well. Let $Q_z(\ell) = \lim_{k\to\infty} Q_z(\ell,k)$, $\ell = 0, 1, 2, ...$ Taking the limit in (21) we have

$$Qq \triangleq \begin{bmatrix} Q_z(1) \\ \vdots \\ Q_z(q) \end{bmatrix} = \begin{bmatrix} CA\ Q_x C^T \\ \cdots \\ CA^q Q_x C^T \end{bmatrix} = \Omega q\ A\ Q_x\ C^T \tag{22}$$

or

$$Q_x C^T = A^{-1} \bar{\Omega}_q\ Qq \tag{23}$$

According to the ergodic property of z(i), we can replace the $Q_z(\ell)$ by their sample correlation matrices

$$\hat{Q}_z(\ell) = \frac{1}{N} \sum_{k=1}^{N} Z(k+\ell)\ Z^T(k) \qquad = 0, 1, . . ., q \tag{24}$$

to form matrix Qq. Thus $Q_x C^T = A^{-1}\ \bar{\Omega}_q\ Qq$, or $\hat{A}^{-1}\ \hat{\bar{\Omega}}_q\ \hat{Q}q$ when A is estimated. From the first of equations (21) the estimate for R becomes

$$\hat{R} = \hat{Q}_z(0) - C\ \widehat{Q_x C^T} \tag{25}$$

A second special case occurs when the dimension of the plant noise is less than or equal to the number of independent observations, p≤m. Then matrix Q can be obtained as follows. From the first of equations (21), and noting that $Q_x(k)\to Q_x$, $Q_z(o,k)\to Q_z(o)$ as $k\to\infty$,

$$Q_z(o) - R = C\ Q_x\ C^T \tag{26}$$

From equation (1), $Q_x(k+1) = A\ Q_x(k)A^T + \Gamma Q \Gamma^T$, and upon taking the limit $Q_x(k+1)$ and $Q_x(k)$ are replaced by Q_x to obtain

$$\text{vec}\ Q_x = (I_{n^2} - A\otimes A)^{-1}(\Gamma\otimes\Gamma)\ \text{vec}\ Q \tag{27}$$

Substitution into (26) and replacing $Q_z(o)$ by $\hat{Q}_z(o)$, R by $\hat{R}$, and (if appropriate A by $\hat{A}$)

$$\text{vec}(\hat{Q}_z(o) - \hat{R}) = (C\otimes C)(I_{n^2} - A\otimes A)^{-1}(\Gamma\otimes\Gamma)\ \text{vec}\ Q \tag{28}$$

from which vec Q can be determined.

IDENTIFICATION OF THE SYSTEM MATRIX

Various methods for system identification exist [5,6]. Here we present a simple method for explicit identification of A which uses the observation correlation, and involves only the solution of a set of linear equations. The method is useful for the identification problem associated with an nth order scalar differential equation with any m n independent measurements of the state. The method also applies to general multivariable systems when m=n. The approach used here has some of the spirit of [6], but is fundamentally different, and results in simpler equations.

Equation (6) can be converted into a multivariate time series form

$$Z(k) = \Omega q A \Omega \bar{q}\, Z(k-1) + \Omega q A \Omega \bar{q} \Gamma w(k-1)$$
$$+ V^*(k) - \Omega q A \Omega \bar{q}\, V^*(k-1) \tag{29}$$

Since $V^*(k)$ consists of $w(k), \ldots, w(k+q-1)$ and $v(k), \ldots, v(k+q)$, $Z(k)$ has $(q+1)$- correlation. Hence

$$E\left[Z(k)Z^T(k-q-2)\right] = \Omega q A \Omega \bar{q}\, E\left[Z(k-1)Z^T(k-q-2)\right] \tag{30}$$

which can be solved for $M_1 \triangleq \Omega q A \Omega \bar{q}$. In a similar manner one can generate $Mj \triangleq \Omega q A^j \Omega_{\bar{q}}$ as

$$M_j = E\left[Z(k)Z^T(k-q-j-1)\right] E\left[Z(k-j)Z^T(k-q-j-1)\right]^{-1} \tag{31}$$
$$j = 1, 2, \ldots, n$$

For convenience, define

$$Mo \triangleq \begin{bmatrix} I_n & 0 \\ 0 & 0 \end{bmatrix}_{mq \times mq} \tag{32}$$

It is assumed that the system is cyclic so that matrix A can be expressed in companion form

$$A = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ & & & \ddots & \\ 0 & 0 & 0 & \cdots & 1 \\ a_1 & a_2 & a_3 & \cdots & a_n \end{bmatrix} \tag{33}$$

Then according to the Caley-Hamilton theorem one can write

$$a_1 Mo + a_2 M_1 + \ldots + a_n M_{n-1} = M_n \tag{34}$$

or equivalently

$$[\text{vec } Mo, \text{vec } M_1, \ldots, \text{vec } M_{n-1}]\, [a_1, \ldots, a_n]^T = \text{vec } M_n \tag{35}$$

which represents a set of $(mq)^2$ linear equations. Furthermore, these equations have a unique solution under the assumption that the system (1) and (2) is observable.

It is assumed that $w(k)$ and $v(k)$ are ergodic and that all eigenvalues of $A \otimes A$ lie inside the unit circle, which means that $Z(k)$ and $X(k)$ are asymptotically stationary and ergodic. Hence, we can use the sample correlation in place of the expectations of equation (31). Define M_j and a_j such that

$$\widehat{M}_j(N) \triangleq \left[\sum_{i=1}^{N} Z(i)Z^T(i-q-j-1)\right] \left[\sum_{i=1}^{N} Z(i-j)Z^T(i-q-j-1)\right]^{-1} \tag{36}$$
$$j = 1, 2, \ldots, n$$
$$\widehat{Mo} = Mo$$

$$[\text{vec}\widehat{Mo}, \text{vec}\widehat{M}_1, \ldots, \text{vec } \widehat{M}_{n-1}][\hat{a}_1, \ldots, \hat{a}_n]^T = \text{vec}\widehat{M}_n \tag{37}$$

Equation (37) is the fundamental equation which gives the estimate $\hat{A}$ of A by solution of a set of linear equations.

Note that in the special case where m=n, an alternative procedure applies. The $\Omega q = C$ and $\Omega\bar{q} = c^{-1}$, and without requiring A to have companion form, it can be estimated by

$$A = C^{-1}[\sum_{i=1}^{N} Z(i)Z^T(i-2)][\sum_{i=1}^{N} Z(i-1)Z^T(i-2)]^{-1}C \tag{38}$$

ASYMPTOTIC PROPERTY OF THE ESTIMATORS

In order to establish the asymptotic property of the estimators presented previously, we need the martingale convergence result shown in the following Lemma.

Lemma. Let ε be a martingale difference sequence, i.e. $E(\varepsilon_i | \mathcal{F}_{i-1}) = 0$ and $E(\varepsilon_i^2 | \mathcal{F}_{i-1}) = \sigma^2$ where $\{\mathcal{F}_k\}$ is an increasing σ-field sequence generated by $\{\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_k\}$. Let U_i be an $\mathcal{F}_{i-1}$ - measurable sequence and asymptotically stationary. Then

$$\frac{1}{N}\sum_{i=1}^{N} U_i \varepsilon_i \to 0 \quad \text{a.s. as } N \to \infty \tag{39}$$

Proof. Because U_i is asymptotically stationary, $EU_i \to Qu$ as $i \to \infty$. Thus

$$\sum_{i=1}^{\infty} E(U_i^2/i^2) = \sum_{i=1}^{\infty} \frac{1}{i^2} EU_i^2 < \infty$$

$$E\sum_{i=1}^{\infty} U_i^2/i^2 = \sum_{i=1}^{\infty} E(U_i/i^2) < \infty$$

and $\sum_{i=1}^{\infty} U_i \varepsilon_i < \infty$, a.s. From the assumption of the Lemma we have

$$E[\sum_{i=1}^{\infty}(U_i^2\varepsilon_i^2/i^2) | \mathcal{F}_{i-1}] = \sum_{i=1}^{\infty}(U\ /i^2)E(\varepsilon_i^2 | \mathcal{F}_{i-1}) < \infty$$

It is easy to see that $\{U_i\varepsilon_i/i\}$ is a martingale. According to Doob and Chow's local martingale convergence theorem we have $\sum_{i=1}^{\infty} U_i \varepsilon_i/i < \infty$ a.s., and then by using the Kronecker Lemma this implies $(\sum_{i=1}^{N} U_i\varepsilon_i)/N \to 0$ a.s., as $N \to \infty$

This result can be generalized immediately to handle the multivariante case. Next, consider the strong consistent property of $\hat{A}(N)$.

Theorem. Let the $\hat{M}j(N)$, $j=1, 2, \ldots, n$, of equation (36) be the estimators of Mj in (31). If $A \otimes A$ has all its eigenvalues within the unit circle, and w(k) and v(k) are ergodic, then $\hat{M}i\ (N) \to Mi$, a.s., as $N \to \infty$, and therefore $\hat{A}(N) \xrightarrow{a.s.} A$. In addition, $\hat{Q}(N) \xrightarrow{a.s.} Q$ and $\hat{R}(N) \xrightarrow{a.s.} R$ as $N \to \infty$.

Proof. According to the assumptions, z(k) is asymptotically stationary and ergodic, and hence Z(k) is as well. Making use of (29)

$$\frac{1}{N}\sum_{i=1}^{N} Z(i-1)Z^T(i-1-2) \xrightarrow{a.s.} P(q+1) \qquad \text{as } N \to \infty$$

$$\frac{1}{N}\sum_{i=1}^{N} Z(i)Z^T(i-q-2) = \frac{1}{N}\sum_{i=1}^{N} M_1 Z(i-1)Z^T(i-q-2)$$

$$+ \frac{1}{N}\sum_{i=1}^{N} M_1\Gamma w(i-1)Z^T(i-q-2) + \frac{1}{N}\sum_{i=1}^{N} V^*(i)Z^T(i-q-2)$$

$$- \frac{1}{N}\sum_{i=1}^{N} M_1 V^*(i-1)Z^T(i-q-2)$$

The first term on the right hand side tends to $M_1P(q+1)$ (a.s.) and the other three terms tend to zero (a.s.) as $N \to \infty$, since every element in the last three terms has the structure discussed in the Lemma. Therefore, $M_1(N)$

$\longrightarrow M_1$, a.s., as $N \to \infty$. Similar reasoning establishes the desired result for Mj.

Since $\hat{a}_1, \hat{a}_2, \ldots, \hat{a}_n$ can be solved from the linear equation (37), $(\hat{a}_1, \hat{a}_2, \ldots, \hat{a}_n) \xrightarrow{a.s.} (a_1, a_2, \ldots, a_n)$ as $N \to \infty$. Similar reasoning shows that the $\hat{A}(N)$ in (38) satisfies $\hat{A}(N) \xrightarrow{a.s.} A$ as $N \to \infty$ as well.

Based on the ergodic assumption $\hat{P}(N,0)$ and $\hat{P}(N,q+1)$ of equations (16) tend to P(0) and P(q+1) (a.s.) as $N \to \infty$, and hence $\hat{R}^N_{v*} \xrightarrow{a.s.} Rv*$ as $N \to \infty$. Noting that $\hat{R}(N) = Rv*\ (1,1)$ and that $\mathrm{vec}\hat{Q}(N)$ can be solved from linear equations, it is clear that $\hat{R}(N) \xrightarrow{a.s.} R$ and $\hat{Q}(N) \xrightarrow{a.s.} Q$ as $N \to \infty$. If A is estimated as well, noting that $\hat{A}(N)$ is consistent, and noting the simple relation between Ωq and $\hat{\Omega}q$, we see that the strong consistent property of $\hat{R}$ and $\hat{Q}$ still holds.

Remark. Note that the form of the estimate Mj(N) in (36) is similar to a least square estimator, and that the computation of A(N) can be converted to a recursive form using

$$\begin{aligned} \hat{M}j(N) &= \hat{M}j(N-1)+[Z(N)-\hat{M}j(N-1)Z(N-j)]K_N \\ K_N &= Z^T(N-j)P_{N-1}/[1+Z^T(N-q-2)P_{N-1}Z(N-j)] \\ P_N &= P_{N-1}-P_{N-1}Z(N-j)Z^T(N-q-2)P_{N-1}/[1+Z^T(N-q-2)P_{N-1}Z(N-j)] \end{aligned} \tag{40}$$

and that $\hat{R}(N)$ and $\hat{Q}(N)$ can also be calculated recursively using

$$P(N,\ell) = P(N-1,\ell) + \frac{1}{N}[Z(N-\ell)Z^T(N)-P(N-1,\ell)] \tag{41}$$

$$\ell = o,\ q+1$$

These recursive equations have applications in adaptive filtering.

REFERENCES

1. R.K. Mehra, "On the Identification of Variances and Adaptive Kalman Filtering," IEEE Transactions on Automatic Control, AC 15, No. 2, 1970, pp. 175-184.

2. R.K. Mehra, "Approaches to Adaptive Filtering," IEEE. Tran. on Automatic Control, Vol. AC-17. No. 5, 1972, pp. 693-698.

3. K.A. Myers and B.D. Tapley, "Adaptive Sequential Estimation with Unknown Noise Statistics," IEEE Trans. Automatic Control, Vol. AC-21, 1976, pp. 520-523.

4. J.F. Leathrum, "On Sequential Estimation of State Noise Variances," IEEE Trans. Automatic Control, Vol. AC-26, 1981, pp. 745-746.

5. G.C. Goodwin and R.L. Payne, Dynamic System Identification, Academic Press, New York, 1977.

6. R.K. Mehra, "On-line Identification of Linear Dynamic Systems with Applications to Kalman Filtering," IEEE Transactions on Automatic Control, Vol. AC-16, No. 1, 1971.

7. H. Neudecker, "Some Theorems on Matrix Differentiations with Special Reference to Kronecker Matrix Products," J. Amer. Stat. Assoc., Vol. 64, 1969, pp. 953-963.

A BILINEAR OBSERVER APPROACH TO A CLASS OF NON-LINEAR STATE RECONSTRUCTION PROBLEMS

HEBERTT SIRA-RAMIREZ
Control Systems Department, Universidad de Los Andes, Merida-Venezuela.

ABSTRACT

A class of deterministic non-linear state reconstruction problems generated by a linear dynamic plant with non-linear state-output maps is considered. For a large class of output non-linearities it is shown that the problem can be reduced to a deterministic bilinear state reconstruction problem for which a number of results are already available [1]-[5].

PROBLEM FORMULATION AND MAIN RESULTS

Consider a deterministic state reconstruction problem defined on :

$$\dot{\underline{x}}(t) = A\,\underline{x}(t) + B\,\underline{u}(t) \tag{1}$$

with $\underline{x} \varepsilon R^n$, $\underline{u} \varepsilon R^m$, $B = [\underline{b}_1, \underline{b}_2, \ldots, \underline{b}_m]$ and a non-linear output of the form:

$$\underline{y} = \underline{g}(\underline{x}) = \tilde{G}_p\, \tilde{\underline{x}}^{[p]} \tag{2}$$

where $\tilde{\underline{x}}^{[p]}$ is the p-th homogeneous tensor power of $[{1 \atop \underline{x}}]$ known as the p-th family of tensor powers of $\underline{x}$; $(\tilde{\underline{x}}^{[p]})' = [1, \underline{x}', (\underline{x}^{[2]})', \ldots, (\underline{x}^{[p]})']$. $\underline{x}^{[k]}$ has components constituted by the lexicographic listing of the $N(n,k) = \binom{n+k-1}{k}$ linearly independent forms $x_1^{k}1 x_2^{k}2 \ldots x_k^{k}n$ with $\Sigma k_i = k$.

This problem requires, in general, a non-linear observer [1] with all the hardware and theoretical difficulties involved [2].

We consider the possibility of reducing the above problem to an equivalent bilinear state reconstruction problem for which a good deal of results are known. This possibility is easily established from the fact that the input-output behaviour of (1)-(2) is totally equivalent to that of :

$$\frac{d}{dt}\tilde{\underline{x}}^{[p]} = \left(\tilde{A}_{[p]} + \sum_{i=1}^{m} u_i\, \tilde{B}_{i[p]} \right) \tilde{\underline{x}}^{[p]} \tag{3}$$

$$\underline{y} = \tilde{G}_p\, \tilde{\underline{x}}^{[p]} \tag{4}$$

where u_i is the i-th component of $\underline{u}$, $\tilde{A}_{[p]} = \mathrm{diag}[0, A, A_{[2]}, \ldots, A_{[p]}]$; $\tilde{B}_i = \begin{vmatrix} 0 & 0 \\ \underline{b}_i & 0 \end{vmatrix}$ and $\tilde{B}_{i[p]}$ has its non-zero entries in blocks immediately below the main diagonal blocks with dimensions corresponding to the tensor power components of $\tilde{\underline{x}}^{[p]}$. $\underline{b}_i$ is the i-th column of B.

Proposition Let $\tilde{G}_p = [G_0, G_1, \ldots, G_p]$, then the pairs $(A_{[k]}, G_k)$ are observable for all k if the pair $(\tilde{A}_{[p]}, \tilde{G}_p)$ is observable.

It is also simple to show that the Lie Algebra generated by the structural matrices of (3) is solvable thus existing the possibility of upper-triangularization of the systems equations and a global representation for the solution as products of matrix exponentials.

An observer for (3)-(4) is readily given by :

$$\frac{d}{dt}\underline{z} = \left(\tilde{A}_{[p]} + \sum_{i=1}^{m} u_i \tilde{B}_{i[p]}\right) \underline{z} + H_0(\underline{y} - \underline{w}) + \sum_{i=1}^{m} u_i H_i (\underline{y} - \underline{w}) \tag{5}$$

$$\underline{w} = \tilde{G}_p\, \underline{z} \tag{6}$$

The estimation error $\underline{e} = \underline{z} - \tilde{\underline{x}}^{[p]}$ is governed by the dynamics:

$$\frac{d}{dt}\underline{e} = (\tilde{A}_{[p]} - H_0 \tilde{G}_p)\,\underline{e} + \sum_{i=1}^{m} u_i (\tilde{B}_{i[p]} - H_i \tilde{G}_p)\,\underline{e} \tag{7}$$

Theorem An assymptotic state observer for (3)-(4) (i.e (1)-(2)) exists if and only if for some given $Q = Q' > 0$ there exists a $P = P' > 0$ such that:

$$P(\tilde{A}_{[p]} - H_0\tilde{G}_p) + (\tilde{A}_{[p]} - H_0\tilde{G}_p)'P + Q = 0 \qquad (8)$$

$$P(\tilde{B}_{i[p]} - H_i\tilde{G}_p) + (\tilde{B}_{i[p]} - H_i\tilde{G}_p)'P = 0 \; ; \; i=1,2,\dots,m \qquad (9)$$

If the pair $(\tilde{A}_{[p]},\tilde{G}_p)$ is observable, condition (8) is automatically satisfied for a suitable matrix H_0 which places the poles in the left halfplane. This requirement can also be weakened to have $(\tilde{A}_{[p]},\tilde{G}_p)$ detectable.

It is also easy to see that a necessary condition for the existence of a non-trivial solution to equations (9), $(\tilde{B}_{i[p]} - H_i\tilde{G}_p)$ must have either a zero eigenvalue or else a pair of opposite eigenvalues (w.r.t the origin)

CONCLUSIONS AND SUGGESTIONS FOR FURTHER RESEARCH

We have examined a class of non-linear state reconstruction problems defined on linear plants with non-linear multinomial outputs. A higher dimensional bilinear state estimation problem, requiring simpler hardware for analog simulation or simpler digital computer software, can be proposed for the solution of the original problem. A direct non-linear approach for the class of output maps considered requires an extremely large collection of multipliers and ,generally, sophisticated analog computer capabilities for implementation or design experiments. The conditions we propose for the existence of an assymptotic observer are represented by Lyapunov algebraic equations whose solution can be obtained by standard computer routines extensively available.

The ideas and results in [6] and [7] allow extension of these results to the case of non-linear analytic output maps expressible, in an approximate fashion, by a suitable linear transformation of a conveniently high degree family of homogeneous tensor powers of the state vector.

A Lie Algebraic setting for the bi-linear state reconstruction problem, and for the class of problems here presented, require special attention as a possible research area.

REFERENCES

[1] Kou,S.R., Elliot,D.L., and Tarn,T.J., "Exponential Observers for Non-linear Dynamic Systems," Information and Control, 29, pp 204-216. 1975.

[2] Zeits, M., " Nichtlineare Beobachter," Regelungstechnik, Vol. 27, pp 241-249. 1979.

[3] Hara, S., and Furuta, K., " Minimal Order Observers for Bilinear Systems," International Journal on Control. Vol. 24, No. 5, pp 705-718. 1976.

[4] Funahashi, Y., "Stable State Estimator for Bilinear Systems," International Journal on Control. Vol. 29, No.2, pp 181-188. 1979.

[5] Derese,J., and Noldus,E.J., " Non-linear Control of Bilinear Systems," IEE Proceedings, Vol. 27, Part D. No.4 , July 1980.

[6] Loparo,K., and Blankenship,G.,"Estimating the Domain of Attraction of Non-linear Feedback Systems," IEEE Transactions on Automatic Control Vol. AC-23, No. 4, pp 602-608. August 1978.

[7] Sira-Ramirez, H.,"Determining the Feasible Region for Non-stationary Non-linear Feedback Systems," Proceedings XVII Annual Allerton Conference on Communications, Control and Computing. pp 571-577. October 1979.

Acknowledgement The autor wishes to thank the useful cooperation of Mr. Eliecer Colina at Case Western Res. Univ. for providing key references.

OBSERVER METHODS IN SELF-TUNING CONTROL

K. WARWICK
Dept. of Electrical Engr.
Imperial College
London SW7, ENGLAND

P. J. ANTSAKLIS
Dept. of Electrical Engr.
University of Notre Dame
Notre Dame, IN 46556, USA

ABSTRACT

A state-space formulation of the self-tuning control problem has been recently used to develop new self-tuning control algorithms and to study existing ones developed using the polynomial representation. A number of state estimation techniques used in self-tuning control are examined using the above formulation and it is shown that only certain recent self-tuning methods [7,6] employ an optimal state estimator.

SUMMARY

Self-tuning theory is used in the control of systems the parameters of which can be unknown and may vary with respect to time. Much of the earlier work, e.g. Åström & Wittenmark (1973), Clarke & Gawthrop (1975), was developed using a representation involving polynomials in the backward shift (delay) operator; in the last few years, however, a state-space formulation has been introduced, Warwick (1981), Tsay and Shieh (1981), which has not only allowed for a wider choice of possible control actions, but has also provided an insight into the theory underlying the self-tuning methods developed using the polynomial description.

As, generally, only plant input and output measurements may be obtained at each sampling interval, the state vector, necessary in the state-space design philosophy, must be estimated so that it can be formed from a combination of the input and output regressive measurements, and the plant parameter estimates. This, however, leads to the possible use of one of a large number of observer schemes; the most popular self-tuners employ standard observer forms, as found in Kwakernaak & Sivan (1972), and the state estimator is generally obtained by considering the plant parameters to be known, the fact that they may well not have been identified being taken into account later.

It is shown here, that certain schemes, e.g. Warwick (1981) and Tsay and Shieh (1981), employ an observer which is optimal (the Kalman-Bucy equivalent) in the sense that it attempts to minimize the mean square error in state reconstruction. This observer also contains the noise coloring polynomial as a factor of its characteristic polynomial, thus enabling a reasonably simple study of both the stability and the reconstruction error properties of the observer to be carried out. One property is immediately apparent, however, namely, the fact that if the disturbance is white noise then the observer becomes a dead-beat observer.

The noise coloring polynomial appears in the characteristic polynomial of the observer in several other self-tuning controllers, e.g. in the state-space equivalent form of (see Lam 1980) Clarke & Gawthrop (1975) and Wellstead et al. (1979). However, due to the formulation of the estimator equation to include the present measured output, the observer must be regarded in this case as being non-optimal in the same sense as previously mentioned. The different estimates of the state vectors can, though, be related via a simple theorem and hence by designing the control law on the certainty equivalence principle, the various controller actions decided upon in this case can be expressed in terms of the optimal state estimate by means of a straightforward modification.

Of the more recently developed self-tuning controllers, several are of interest: In Wellstead and Sanoff (1981), the observer polynomial is essentially specified by the designer, and does not necessarily coincide with the disturbance polynomial; the effect this has on the reconstruction error is discussed. The varying characteristic polynomial inherent when linear output feedback is employed in combination with state feedback, Warwick and Westcott (1982), is also discussed, as well as the use of a state vector constructed solely from regressive measured values, Hesketh (1982).

REFERENCES

[1] Åström K. J. & Wittenmark B., "On Self-Tuning Regulators", Automatica, Vol. 9, pp. 185-199, 1973.

[2] Clarke D. W. and Gawthrop P.J., "Self-Tuning Controller", Proc. IEE, Vol. 122, No. 9, pp. 929-934, 1975.

[3] Hesketh T., "State-Space Pole Placing Self-Tuning Regulator using Input-Output Values", Proc. IEE, Pt. D, pp. 123-128, 1982.

[4] Kwakernaak H. and Sivan R., Linear Optimal Control Systems, Wiley-Interscience, New York, 1972.

[5] Lam K. P., "Implicit and Explicit Self-Tuning Controllers", D. Phil. Thesis, OUEL Report 1334/80, Oxford University, 1980.

[6] Tsay Y. T. and Shieh L.S., "State-Space Approach for Self-Tuning Feedback Control With Pole Assignment", Proc. IEE, Pt. D, pp. 93-101, 1981.

[7] Warwick K., "Self-Tuning Regulators - A State-Space Approach", Int. J. Control, Vol. 33, No. 5, pp. 839-858, 1981.

[8] Warwick K. and Westcott J. H., "Arbitrary Linear Output Feedback Self-Tuners", Proc. 6th IFAC Symposium on Identification and System Parameter Estimation, Washington, D.C., pp. 1040-1045, 1982.

[9] Wellstead P. E., Prager D. and Zanker P., "Pole Assignment Self-Tuning Regulator", Proc. IEE, Vol. 126, pp. 781-787, 1979.

[10] Wellstead P. E. and Sanoff S. P., "Extended Self-Tuning Algorithm", Int. J. Control, Vol. 34, No. 3, pp. 433-455, 1981.

A DIGITAL COMPENSATION TECHNIQUE FOR PRESSURE TRANSDUCERS

C. E. (MIKE) LINDAHL
Micro Switch/a Honeywell division
11 W. Spring Street
Freeport, IL 61032

ABSTRACT

Typical silicon pressure sensors exhibit nonlinear response to temperature and pressure effects. This paper describes a technique for linearity and temperature compensation via a microprocessor.

High accuracy can be achieved by modeling and removing the predictable error terms, namely linearity and temperature shift. A second order model can be used to obtain a desired 0.9% accuracy. This can be accomplished by conversion of the sensor output to digital form computation of the corrections and either digital data transmission or reconversion to analog form. The computation coefficients are programmed in ROM during characterization of the transducer. Scaling for different applications can also be accommodated in the microprocessor based design.

This paper discusses the design considerations for a digital compensation technique for pressure transducers.

INTRODUCTION

The output accuracy of a silicon pressure sensor chip can be improved by software with a microprocessor. The silicon pressure sensor chip exhibits nonlinear response to temperature and pressure variations. By the use of a microprocessor, with the mathematical coefficients of the silicon pressure sensor chip programmed in PROM, the nonlinear response can be compensated.

SILICON PRESSURE SENSORS

The MICRO SWITCH silicon pressure sensor chip is 0.1 inch square with a sensing diaphragm and four ion implanted piezoresistors. The piezoresistors are connected as active elements in a full Wheatstone bridge (Figure 1). With an excitation voltage, Vs, supplied to the pressure sensor, the output voltage, Vo, is a function of the change in resistance of the piezoresistors to a change in pressure (Figure 2).

Since it is a semiconductor device, the silicon pressure sensor exhibits output voltage variation with temperature. The output voltage at null (zero pressure) may either increase or decrease in value with a change in temperature . This null shift is unpredictable, in magnitude and polarity, from one sensor to another. However, the voltage variation at maximum pressure (span voltage) is predictable in that it always decreases with increasing temperature and increases with decreasing temperature . Span voltage error is also referred to as sensitivity error.

Another source of error in a silicon pressure sensor is the nonlinearity of the output voltage curve. The output voltage curve can be C-shaped, S-shaped, or reverse C-shaped . The output voltage curve is unpredictable from one sensor to another.

CHARACTERISTIC EQUATION OF THE SENSOR

Both null shift and sensitivity error can be compensated to ±1% of full scale output (FSO) by analog means. However, it is difficult to achieve null shift and sensitivity errors of ±0.1% FSO due to the inconsistencies of non-linear compensation and the economics of production. A general characteristic equation of the silicon pressure from its pressure and temperature characteristics is,

$$F(P,T) = f_1(P) \cdot f_2(T) + f_3(T) \qquad (1)$$

where, $f_1(P)$ is the change in output due to pressure change.
$f_2(T)$ is the change in sensitivity due to temperature change.
$f_3(T)$ is the change in null due to temperature change.

By rewriting equation (1) as a second order polynomial to fit the sensor's output curve,

$$F(P,T) = a(T)P^2 + b(T)P + c(T) \qquad (2)$$

where, $a(T) = X_{11}T^2 + X_{12}T + X_{13}$
$b(T) = X_{21}T^2 + X_{22}T + X_{23}$
$c(T) = X_{31}T^2 + X_{32}T + X_{33}$

$F(P,T)$, $a(T)$, $b(T)$, $c(T)$ and X_{11} to X_{33} are either known parameters or can be determined by characterization of the silicon pressure sensor chip. From equation (2), $c(T)$ is a function of the null shift of the silicon sensor chip and can be determined by least squares curve fitting of the null shift vs. temperature data. However, $a(T)$ and $b(T)$ must be found by iterative means.

CONCLUSION

A prototype digital compensated pressure transducer based on an Intel STK-86 16-bit microprocessor has been built at MICRO SWITCH. The pressure and temperature data are input through an Intersil ICL7109 12-bit analog to digital converter (Figure 3). By use of the characteristic equation of the silicon pressure sensor chip a desired accuracy of 0.1% FSO can be obtained.

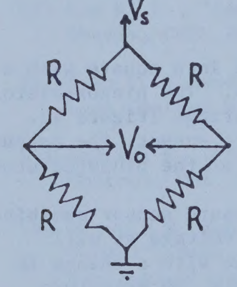


Fig. 1 Four element wheatstone bridge

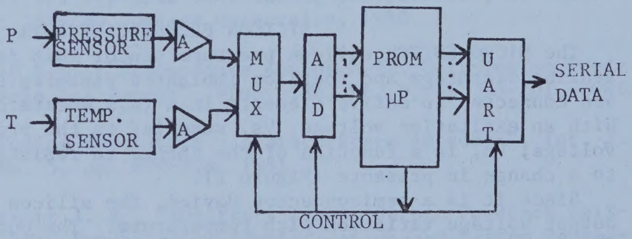


Fig. 3 Block diagram of pressure transducer

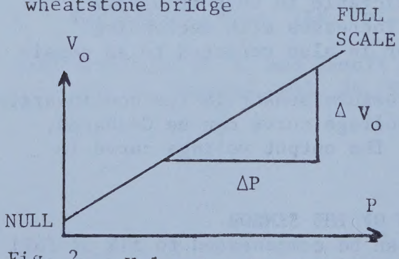


Fig. 2 Voltage output versus pressure input curve

References: Daniel, C. and Wood, F.S., "Fitting Equations to Data, Wiley, 1980. "The 8086 Users Manual", Intel.

STOCHASTIC RELIABILITY FUNCTIONS FOR FAILURE RATES DERIVED FROM GAUSS-MARKOV PROCESSES AND FILTERS WITH POINT PROCESS OBSERVATIONS

JOSEPH L. HIBEY
University of Notre Dame
Notre Dame, IN 46556

ABSTRACT

An extension of the well-known Cameron-Martin formula can be interpreted as the expectation of a stochastic reliability function applicable in those situations where nondecreasing failure rates are desired. This follows if the failure rate is modeled as the square of a Gauss-Markov process. We describe the methodology for the general vector case, and then specialize the results to the one-dimensional case so as to obtain an exact, closed-form expression for the reliability function. Using the theory of recurrent and transient processes, we then show how the choice of a model parameter and the initial state influence reliability. Finally, in the general case, we briefly indicate how filtering theory applied to stochastic point processes can be used to generate estimates for failure rates modeled as Markov processes.

1. Introduction

In the study of both hardware and software reliability, one is sometimes faced with having to choose a mathematical model for the underlying failure rate mechanism. The literature on this subject is replete with both deterministic and stochastic models. Some models of each are given by Komoriya et al. [9] in the context of software reliability for the nuclear power industry. There the authors show that a stochastic model proposed by Littlewood (and Verrall) [11] results in a reliability function that is more conservative than any of the deterministic models considered. Since such conservatism is a highly desirable design objective, we are thereby motivated to study other stochastic models.

We have chosen to model the failure rates as Markov processes because of their physical appeal and the highly developed mathematical theory that exists to describe their behavior. In fact, the majority of this paper is concerned with the study of a particular failure rate model derived from a Gauss-Markov process. Such a model allows us to derive an exact, closed-form expression for the expectation of the reliability function. Furthermore, using the theory of stochastic differential equations as given by Friedman [5, Chap. 9], we are able to study some of its properties and thus describe some potential applications.

Another reason for modeling failure rates as Markov processes is because we are able to generate minimum-mean-squared-error (MMSE) estimates using filtering theory. Martz et al. [12] have applied Kalman filtering techniques in this context. A more natural approach, we feel, is the use of filters derived from observing counting processes. This is the subject of Section 5 of this paper and will include references for several such filters that have appeared in the literature in recent years.

The paper is organized in the following manner. Section 2 contains a general result that represents an extension of the well-known Cameron-Martin formula and, as such, is of independent interest; its proof is outlined in the Appendix. In Section 3, we discuss reliability theory and present a Gauss-Markov model that will be used to derive a failure rate. A scalar version of the results in Section 2 is then used to obtain an exact expression for the average reliability. In Section 4, we apply to our model some results from the one-dimensional theory of recurrent and transient solutions of stochastic differential equations to show how the choice of a model parameter and the initial state influence reliability. Proceeding to Section 5, we briefly discuss the filtering theory of stochastic point processes and then conclude by suggesting some possible filters that have appeared in the literature. These will include both continuous- and discrete-time models.

2. Extension of the Cameron-Martin Formula

Consider the Cameron-Martin formula as given by Lipster and Shiryayev [10, p. 280]:

$$E \exp [- \int_{0}^{T} w_t{}' \, Q_t \, w_t \, dt] = \exp [\, \frac{1}{2} \int_{0}^{T} \mathrm{tr}\, (\Gamma_t) \, dt], \tag{1}$$

where w_t is a standard vector Brownian motion process, ' denotes transpose, Q_t is a symmetric nonnegative definite matrix, tr denotes the trace operator, and Γ_t is a symmetric nonpositive definite matrix uniquely satisfying the Riccati equation

$$\frac{d\Gamma_t}{dt} = - \Gamma_t{}^2 + 2 \, Q_t , \quad \Gamma_T = 0.$$

We wish to generalize this result by considering the evaluation of

$$E \exp [- \int_{t_0}^{T} x_t{}' \, Q_t \, x_t \, dt]$$

where exp [ ] is a so-called multiplicative functional of the vector (Gauss-Markov) Ito process x_t satisfying the nonstationary linear stochastic differential equation

$$dx_t = A_t \, x_t \, dt + B_t \, dw_t , \; x_{t_0} \triangleq x \tag{2}$$

with A and B time-varying matrices of appropriate dimension. Discussion of this type of functional for the case when x_t is a general Markov process is given by Dynkin [3, Chap. 9]. If we define

$$q(t_0 , x) \triangleq E \, [\exp(- \int_{t_0}^{T} x_t{}' \, Q_t \, x_t \, dt) | x_{t_0} = x] \tag{3}$$

then we have the following result:

Theorem: If x_t satisfies (2), then $q(t_0,x)$ in (3) satisfies

$$E_x[\exp(- \frac{1}{2} \, x'\Gamma_{t_0} \, x) \; q(t_0,x)] = \exp[\, \frac{1}{2} \int_{t_0}^{T} \mathrm{tr}\, (B_t{}' \, \Gamma_t \, B_t) dt] \tag{4}$$

where Γ_t is a nonpositive definite matrix satisfying the Riccati equation

$$\frac{d\Gamma_t}{dt} = -\Gamma_t \, A_t - A_t{}' \, \Gamma_t - \Gamma_t \, B_t \, B_t{}' \, \Gamma_t + 2Q_t , \quad \Gamma_T = 0, \tag{5}$$

and E_x denotes expectation with respect to the distribution of x_{t_0}.

Proof: See Appendix.

3. A Failure Rate Derived from a Gauss-Markov Process

We consider the reliability of a system that functions normally until its first time of failure; see, for example, Gnedenko et al. [6, Chap. 2]. Thus, the reliability function $R(\cdot)$ and failure rate process $\lambda(\cdot)$ are related by

$$R(t_o) = \exp\left[-\int_{t_o}^{T} \lambda_t \, dt\right]$$

and denotes the probability of failure-free operation from time t_o to T. Our interest focuses on situations where λ is a random process dependent on some unobservable Markov process $x(\cdot)$. Considering only the scalar case for purposes of exposition, we then are led to consider the stochastic reliability function

$$R(t_o) \overset{\Delta}{=} \exp\left[-\int_{t_o}^{T} \lambda(x_t) \, dt\right]$$

and its expected value

$$\overline{R}(t_o) \overset{\Delta}{=} E \exp\left[-\int_{t_o}^{T} \lambda(x_t) dt\right] \tag{6}$$

where the expectation is with respect to the probability measure induced by the random process $x(\cdot)$.

The failure rate process $\lambda(\cdot)$ is sometimes characterized in terms of the so-called "bathtub" curve; see Gnedenko [6, p. 85]. Its shape corresponds to three periods of operation; the initial or debugging period where λ is decreasing, the normal period where λ is nearly constant, and the wear-out period where λ is increasing. It is basically the last two periods that we are interested in discussing in this section and the next. Thus we are proposing to model systems that gradually wear-out or deteriorate such as certain hardware components. The model would not be appropriate for systems where decreasing failure rates are required, such as those discussed by Wong [23, pp. 402-407] for electronic equipment, or for systems used to determine software reliability such as those discussed by Littlewood [11] (in the sense that a good working computer code does not deteriorate like some hardware components). This model, then, is of the type of continuous shock and wear processes discussed by Esary et al. [4, pp. 627-649] and Cinlar [1, pp. 193-214], but does not explicitly model the occurrence of shocks.

We now develop our model by considering a scalar, time-invariant version of (2). Thus, with $A_t = a$ and $B_t = 1$,

$$dx_t = ax_t \, dt + dw_t, \quad x_{t_o} = x \tag{7}$$

represents the evolution of the Gauss-Markov process x_t. For the failure rate process $\lambda(x_t)$ appearing in (6), we define $\lambda(x_t) \overset{\Delta}{=} \theta \, x_t^2$, where the constant $\theta > 0$. Therefore, substituting this failure rate into (6) and reconditioning on the initial state x_{t_o} gives

$$\overline{R}(t_0) = E_x \{E[\exp(-\theta \int_{t_0}^{T} x_t^2 \, dt) | x_{t_0} = x]\}. \tag{8}$$

The inner conditional expectation in (8) is just the $q(t_0,x)$ defined in (3) with $Q_t = \theta$. Furthermore, the Riccati equation given by (5) becomes

$$\dot{\Gamma} = -2a\Gamma - \Gamma^2 + 2\theta$$

and its solution is easily found, with $\beta \overset{\Delta}{=} \sqrt{a^2 + 2\theta}$, to be

$$\Gamma_t = \frac{-2\theta[1 - e^{2\beta(t - T)}]}{\beta - a + (\beta + a)e^{2\beta(t - T)}} . \tag{9}$$

Now, if we assume x is deterministic, (8) becomes $\overline{R}(t_0) = q(t_0,x)$ and (4) of the above theorem becomes

$$q(t_0,x) = \exp[\tfrac{1}{2} x^2 \Gamma_{t_0} + \tfrac{1}{2} \int_{t_0}^{T} \Gamma_t \, dt]. \tag{10}$$

Finally, we substitute (9) into (10) and integrate to obtain the average reliability function

$$\overline{R}(t_0,x) = \left[\frac{2\beta e^{(\beta + a)(t_0 - T)}}{\beta - a + (\beta + a)e^{2\beta(t_0 - T)}} \right]^{1/2} \cdot \exp\left\{ \frac{(a^2 - \beta^2)[1 - e^{2\beta(t_0 - T)}]x^2}{2(\beta - a) + 2(\beta + a)e^{2\beta(t_0 - T)}} \right\} \tag{11}$$

where we have written $\overline{R}$ in a way to emphasize its dependence on the initial state x. This dependence of the average reliability function on x as well as the parameter a of (7) is the subject of the next section.

Remark 1: Using the fact that x_T is a Gaussian random variable with known mean and variance, an equivalent expression for $\overline{R}(t_0,x)$ can be shown to be

$$\overline{R}(t_0 x) = \left[\frac{2\beta}{(\beta - a)(e^{2\beta(T - t_0)} - 1) + 2\beta} \right]^{1/2} \cdot \exp\left\{ \frac{\beta(a - \beta)e^{2\beta(T - t_0)}x^2}{(\beta - a)[e^{2\beta(T - t_0)} - 1] + 2\beta} - (a - \beta)(T - t_0 + x^2) \right\} .$$

4. Dependence of Reliability on Model Trajectory

To discuss the dependence of the expectation of the reliability function $R(t_0,x)$ of (11) on the parameter a and initial state x of (7), we use some of the theory of recurrent and transient solutions of stochastic differential equations as given by Friedman [5, Chap. 9]. Definitions of recurrent and transient processes are given on p. 196 of this reference; as stated there, given the initial state x, the process x_t is recurrent if it "visits" any open subset of Euclidean space at a sequence of times increasing to infinity, and the process x_t is transient if it "wanders out to infinity" with probability one. Theorem 7.1 of Friedman [5, pp. 219-220] characterizes the solutions as recurrent or transient in terms of a function $\phi(x)$ (called the scale function by Wong [22, pp. 197-199]) defined as

$$\phi(x) \overset{\Delta}{=} \int_0^x \exp\left[- \int_0^z \frac{2b(u)}{\sigma^2(u)}\, du\right] dz \tag{12}$$

where b is the drift coefficient and σ is the diffusion coefficient in a one-dimensional stochastic differential equation. (Results for multi-dimensional processes are also given). Specifically it states, among other results, the following:

(i) if $\phi(-\infty) = -\infty$ and $\phi(\infty) = \infty$, then the solution ξ_t of a stochastic differential equation described by coefficients b and σ is recurrent; otherwise, the solution is transient.

(ii) if ξ_t has initial state x, then

$$P\{\lim_{t\to\infty} \xi_t = -\infty\} = \frac{\phi(\infty) - \phi(x)}{\phi(\infty) - \phi(-\infty)} ,$$

$$P\{\lim_{t\to\infty} \xi_t = \infty\} = \frac{\phi(x) - \phi(-\infty)}{\phi(\infty) - \phi(-\infty)} ,$$

$$P\{\lim_{t\to\infty} |\xi_t| = \infty\} = 1.$$

Applying these results to our problem, (7) implies that b and σ of (12) are au and 1, respectively. Therefore, it follows that

$$\phi(x) = \int_0^x \exp(-az^2)dz$$

so that the cases $a \leqslant 0$ and $a > 0$ must be considered separately:

Case 1) $a \leqslant 0 \Rightarrow \phi(\pm\infty) = \pm\infty$; therefore, by (i), x_t of (7) is recurrent.

Case 2) $a > 0$; $\phi(x) = \sqrt{\frac{\pi}{a}}\,[\mathrm{erf}(x) - \frac{1}{2}]$

where $\mathrm{erf}(x) \overset{\Delta}{=} \frac{1}{\sqrt{2\pi}\;\bar{\sigma}} \int_{-\infty}^{x} \exp\left(- \frac{z^2}{2\bar{\sigma}^2}\right)dz$

and $\sigma^{-2} \overset{\Delta}{=} \frac{1}{2a} \Rightarrow \phi(\pm\infty) = \pm \frac{1}{2}\sqrt{\frac{\pi}{a}}$; therefore, by (i), x_t of (7) is transient and by (ii),

$$P\{\lim_{t\to\infty} x_t = -\infty\} = 1 - \mathrm{erf}\,(x), \tag{13}$$

$$P\{\lim_{t\to\infty} x_t = \infty\} = \mathrm{erf}\,(x)\ . \tag{14}$$

Recalling that we have modeled the failure rate process λ as $\lambda(x_t) = \theta x_t^2$, we write (6) as

$$\overline{R}(t_0) = E\,\exp[-\,\theta \int_{t_0}^{T} x_t^2\, dt]$$

and draw the following conclusions about this average reliability function as T approaches ∞:

If the model parameter a of (7) is negative, then x_t is recurrent. Thus, by definition of recurrent processes, since x_t will visit any open set infinitely often, the failure rate λ will neither increase nor decrease indefinitely. Such a model, therefore, might be applicable during the normal period of the so-called bathtub curve discussed earlier. If a is zero, then x_t is still recurrent; however, the failure rate then has a mean that increases without bound and therefore would not be a good model for the normal period.
On the other hand if a is positive, then x_t is transient and therefore, by definition, will tend to wander to infinity. The choice of the initial state x and parameter a will, according to (13) and (14), determine the probability of it wandering to either $-\infty$ or $+\infty$, but in any case, the failure rate λ will tend to increase. Thus, such a model might be applicable during the wear-out period of a particular system that exhibits deterioration.

For the general case when the failure rate is a function of a possibly nonlinear Markov process, exact expressions for $\overline{R}(t_0)$ in (6) usually do not exist. (They can, however, be expressed as (8) and represented as solutions to partial differential equations similar to the one appearing in Remark 3 of the Appendix). In these cases, therefore, one might first wish to estimate the failure rate λ and thereby generate an estimate for $\overline{R}(t_0)$. Thus we discuss filtering theory in the next section.

5. Estimating Failure Rates Using Filtering Theory

This section, although somewhat independent of the previous ones, is important when one is interested in estimating failure rates that have been modeled as Markov processes. We are primarily interested in generating minimum-mean-square-error (MMSE) estimates $\hat{\lambda}_t$ of λ_t given by the conditional mean

$$\hat{\lambda}_t \overset{\Delta}{=} E(\lambda_t | n_s,\ 0 \leqslant s \leqslant t)$$

where the observation process n that models the number of failures is a doubly stochastic Poisson process with distribution

$$\Pr\,(n_t - n_s = k) = E\left\{ \frac{[\int_s^t \lambda_\tau \, d\tau]^k}{k!} \exp\left[-\int_s^t \lambda_\tau \, d\tau\right] \right\} .$$

Here, the failure rate λ is allowed to depend on some unobservable Markov process x_t. The filtering theory for this type of process is developed by Snyder [18, Chap. 6]. A general discussion of filtering theory using a martingale approach is presented, for example, by van Schuppen [19]. Before proceeding, however, we note the following.

Starting from (6) with $\lambda_t = \lambda(x_t)$,

$$R(t_0) = E \exp[-\int_{t_0}^{T} \lambda_t \, dt] = E\{E[\exp(-\int_{t_0}^{T} \lambda_t \, dt) | n_s , \; 0 \leqslant s \leqslant t]\}$$

$$\geqslant E \exp[-\int_{t_0}^{T} \hat{\lambda}_t \, dt]$$

where the last step follows from Jensen's inequality and the last expectation is with respect to the probability measure induced by the process n. Thus, using MMSE estimates for λ, we have a lower bound on the average reliability function. The filtering procedure now is as follows.

We define the observation equation by

$$n_t = \int_0^t \lambda_s \, ds + m_t , \; n_0 = 0 \tag{15}$$

where m_t is the martingale associated with the counting process n_t having failure rate λ_t. The process λ_t, or x_t if $\lambda_t = \lambda(x_t)$, denotes the state of our system and satisfies some stochastic differential equation called the state equation; for example (see van Schuppen [19, p. 566]), λ might be modeled as

$$d\lambda_t = a_t \, \lambda_t \, dt + \lambda_t \, dw_t , \; \lambda_0 \tag{16}$$

where a_t is some known function and w_t is standard Brownian motion. The filtered estimate $\hat{\lambda}$ for the system described by (15) and (16) satisfies the filtering equation given by

$$d\hat{\lambda}_t = a_t \, \hat{\lambda}_t \, dt + \sigma_{t-} \, (\lambda,\lambda) \, \hat{\lambda}_{t-}^{-1} \, (dn_t - \hat{\lambda}_t \, dt), \; \hat{\lambda}_0 = E\lambda_0 \tag{17}$$

where the error-covariance

$$\sigma_t(\lambda,\lambda) \overset{\Delta}{=} E[(\lambda_t - \hat{\lambda}_t)^2 | n_s , \; 0 \leqslant s \leqslant t]$$

satisfies a Riccati-type equation involving the third moment of the error $\lambda_t - \hat{\lambda}_t$. The notation t- means "just prior to t"; formally, for example,

$$\hat{\lambda}_{t-}^{-1} \overset{\Delta}{=} \lim_{s \uparrow t} \hat{\lambda}_s^{-1} \quad .$$

As is well-known, this filter is infinite-dimensional because of the appearance of the third moment of error which, in turn, is infinitely coupled to all higher-order moments of error. Thus, in the general case, some approximations are required, and the resulting filters are then called suboptimal.

What we propose to do is simply list references for a few finite-dimensional filters, for both continuous and discrete time, that have appeared in recent years. A Kalman filter has been presented by Martz et al. [12, pp. 165-192], but the observation process used there does not appear to be as natural as our approach where the number of failures n_t given by (15) actually drives the filter given by (17).

Model 1) Linear-Poisson-gamma: Segall [14, pp. 116-120].
2) Constrained linear: Snyder [18, p. 321].
3) Bayesian regression-special case of 2: Synder [18, p. 322].
4) Binomial-beta: van-Schuppen [20, p. 1047].
5) Poisson-gamma: van-Schuppen [20, pp. 1047-1048].
6) Markov chain failure rate: Segall [15, pp. 422-431].

The first three are continuous-time models and the last three are discrete-time models. In all cases, the state equations are linear and lead to finite-dimensional recursive filters.

ACKNOWLEDGEMENT

Part of this work was performed at Argonne National Laboratory while the author was a Faculty Research Participant. The program was administered by the Argonne Division of Educational Programs, and supported by the U.S. Department of Energy, Office of Energy Research, through its University/National Laboratory Cooperative Program.

Thanks also to I. Rhodes for earlier suggesting reference [8].

APPENDIX

Proof of theorem: This result is a special case of the Feynman-Kac formula (see, for example, Williams [21, pp. 157-159]) and can be derived in several ways. We proceed in a way analogous to the one used in arriving at (1) and presented by Hibey [7]. Let (Ω, F, P) denote the probability space on which the process x_t is defined. Define a continuous martingale M_t by

$$M_t \overset{\Delta}{=} \int_{t_o}^{t} x_s' \, \Gamma_s \, B_s \, dw_s$$

such that

$$\exp(M_t/2) \in L^1 \text{ for each } t > 0. \tag{18}$$

Also, define a process Z_t by

$$Z_t \overset{\Delta}{=} \exp(M_t - \frac{1}{2} \langle M \rangle_t)$$

where $\langle M \rangle_t$ is the predictable quadratic variation process associated with the martingale M_t. Then, by a theorem of Kazamaki [8, pp. 597-600], Z_t is a martingale. Therefore, it follows that

$$E(Z_T) = E(Z_t) = 1,$$

that is,

$$E \exp\Big[\int_{t_0}^{T} x_t'\Gamma_t B_t \, dw_t - \frac{1}{2}\int_{t_0}^{T} x_t' \Gamma_t B_t B_t' \Gamma_t x_t \, dt\Big] = 1 \qquad (19)$$

where E denotes expectation with respect to the measure P.

Next, we derive an expression for the first term in the exponential in (19) by applying the Ito differential rule to $x_t' \Gamma_t x_t$ to obtain

$$x_T' \Gamma_T x_T - x_{t_0}' \Gamma_{t_0} x_{t_0} = \int_{t_0}^{T} x_t' \dot{\Gamma}_t x_t \, dt + 2 \int_{t_0}^{T} x_t' \Gamma_t A_t x_t \, dt$$

$$+ \int_{t_0}^{T} \operatorname{tr}(B_t' \Gamma_t B_t) dt + 2 \int_{t_0}^{T} x_t' \Gamma_t B_t \, dw_t . \qquad (20)$$

Substituting $\dot{\Gamma}$ from (5) along with $\Gamma_T = 0$ into (20), and then substituting this result into (19), we obtain

$$E \exp\Big[-\frac{1}{2} x_{t_0}' \Gamma_{t_0} x_{t_0} - \int_{t_0}^{T} x_t' Q_t x_t \, dt\Big]$$

$$= \exp\Big[\frac{1}{2}\int_{t_0}^{T} \operatorname{tr}(B_t' \Gamma_t B_t) dt\Big] . \qquad (21)$$

To conclude the derivation, we use the smoothing property of conditional expectation to recondition the left side of (21) with respect to the initial state $x_{t_0} = x$, and then introduce $q(t_0,x)$ as defined in (3). This completes the proof.

Remark 2: Novikov [13, pp. 717-720] proves that Z_t above is a martingale by requiring that, instead of (18),

$\exp(\langle M\rangle_t/2) \in L^1$ for each $t > 0$. (18')

However, as Kazamaki [8] shows, (18) is an improvement over (18').

Remark 3: Although not directly used in this paper, it is interesting to note that $q(t_0,x)$ satisfies the backward Kolmogorov-type partial differential equation given by

$$-\frac{\partial q}{\partial t_0} = \frac{\partial q}{\partial x} A_{t_0} x + \frac{1}{2} \operatorname{tr}\Big[B_{t_0}' \frac{\partial^2 q}{\partial x^2} B_{t_0}\Big] - x' Q_{t_0} x \, q$$

with boundary condition $q(T,x) = 1$. In fact, starting with this equation, an alternative proof of the theorem can be given.

Remark 4: Results similar to this theorem for stationary processes only were previously derived by other methods in a series of three papers by Darling and Siegert [2] and Siegert [16,17].

REFERENCES

1. Cinlar, E. (1977). Shock and Wear Models and Markov Additive Processes. In The Theory and Applications of Reliability, Vol. 1, eds. C. Tsokos and I. Shimi, Academic Press, New York, 193-214.

2. Darling, D. and Siegert, A. (1957). A Systematic Approach to a Class of Problems in the Theory of Noise and Other Random Phenomena, Part I. IRE Trans. Inform. Theory, IT-3(1), 32-37.
3. Dynkin, E. (1965). Markov Processes, Vol. I, Springer-Verlag, New York.
4. Esary, J., Marshall, A. and Proschan, F. (1973). Shock Models and Wear Processes. Annals of Prob. 1, 627-649.
5. Friedman, A. (1975). Stochastic Differential Equations and Applications, Vol. 1, Academic Press, New York.
6. Gnedenko, B., Belyayev, Y. and Solovyev, A. (1969). Mathematical Methods of Reliability Theory, Academic Press, New York.
7. Hibey, J. (1981). Expectation of a Multiplicative Function of a Linear Ito Process. Intern. Symp. on Inform. Theory, Santa Monica, CA, 22.
8. Kazamaki, N. (1977). On a Problem of Girsanov. Tohoku Math. J., Vol. 29, No. 4, 597-600.
9. Komoriya, H., Mueller, C. and Morris, E. (1981). Quantitative Software Reliability in Nuclear Safety. Proc. Intern. ANS/ENS Topical Meeting on Probabilistic Risk Assessment, October.
10. Lipster, R. and Shiryayev, A. (1977). Statistics of Random Processes I. General Theory, Springer-Verlag, New York.
11. Littlewood, B. (1980). Theories of Software Reliability: How Good Are They and How Can They Be Improved. IEEE Trans. on Software Engr., Vol. SE-6, No. 5, 489-500.
12. Martz, Jr., H., Campbell, K. and Davis, H. (1977). Estimating and Forecasting Failure-Rate Processes by Means of the Kalman Filter. In The Theory and Applications of Reliability, Vol. 1, eds. C. Tsokos and I. Shimi, Academic Press, New York, 165-1982.
13. Novikov, A. (1972). On an Identity for Stochastic Integrals. SIAM Th. Prob. & Applic., Vol. 17, No. 4, 717-720.
14. Segall, A. (1973). A Martingale Approach to Modeling, Estimation and Detection of Jump Processes. Ph.D. Dissertation, Stanford Univ., Stanford, California.
15. ________(1976). Recursive Estimation from Discrete-time Point Processes. IEEE Trans. on Inform. Theory, IT-22(4), 422-431.
16. Siegert, A. (1957), Part II, IRE Trans. Inform. Theory, IT-3(1), 38-43.
17. ________(1958), Part III, IRE Trans. Inform. Theory, IT-4(1), 4-14.
18. Snyder, D. (1975). Random Point Processes, Wiley, New York.
19. van Schuppen, J. (1977). Filtering, Prediction and Smoothing for Counting Process Observations, a Martingale Approach. SIAM J. APPL. MATH., Vol. 32, No. 3, 552-570.
20. ________(1977). Representations and Filtering Problems for Discrete Time Stochastic Processes. Joint Auto. Contr. Conf., San Francisco, CA. 1044-1048.
21. Williams, D. (1979). Diffusions, Markov Processes, and Martingales, Wiley, Chichester.
22. Wong, E. (1971). Stochastic Processes in Information and Dynamical Systems, McGraw-Hill, New York.
23. Wong, K. (1981). Unified Field (Failure) Theory-Demise of the Bathtub Curve. Proc. of Annual Reliability and Maintainability Sym., Phila., PA, 402-407.

CHARACTERIZATION AND ESTIMATION OF TWO DIMENSIONAL ARMA MODELS [*]

R. L. KASHYAP
School of Electrical Engineering
Purdue University
West Lafayette, Indiana 47907

ABSTRACT

A class of two-dimensional autoregressive moving average (ARMA) is introduced having the ability to represent any process with rational spectral density. In this model, the driving noise is correlated and need not be Gaussian. Currently known classes of recursive ARMA models or nonrecursive AR models are subsets of the above class. Next we consider the estimation of parameters of a model to fit a given image using only the empirical correlations and it involves the solution of linear equations.

I. INTRODUCTION

Parametric representations for two-dimensional random fields are useful in many applications like image synthesis, classification, spectral estimation etc. The aim of the paper is to develop a stochastic difference equation model for regular two-dimensional random fields having rational spectral densities and discuss related topics like the various definitions of weak markov processes, parameter estimation and synthesis of textures resembling a given non Gaussian texture.

We will first give the background information regarding the structural representations. Rosanov [1], Woods [2] and Besag [3] have showed that any Gaussian markov field having an all pole spectral density (i.e., a reciprocal of a linear sinusoidal function) possesses a difference equation representation, the so called conditional autoregressive (CAR) model in which the driving input noise is correlated, but does not have, in general, a moving average representation. The set of models suggested earlier such as the simultaneous AR models [3,4], causal recursive models [5,6] is a proper subset of the set of CAR models. However, no general difference equation representation is available for a random field having a rational spectral density, i.e., a ratio of 2 sinusoidals functions. The recursive ARMA model in [7] or the semi causal ARMA model in [8,9] are very restrictive and they cannot model a simple spectral density such as $[1+\phi_1(\cos\lambda_1+\cos\lambda_2)]/[1-\theta_1(\cos\lambda_1+\cos\lambda_2)]$, $|\phi_1|,|\theta_1| < 0.5$, $\phi_1 \neq \theta_1$. This state of affairs is in contrast with the one-dimensional case in which the class of recursive ARMA models can handle any process with rational spectral density.

The structure of the process is intimately connected with the concept of a markov process. The 2D case differs significantly from the 1D case in relation to the

---

[*] Partially supported by Office of Naval Research under grant N00014-82-K-0360 and the Purdue University CIDMAC Project.

markov property. There are 3 types of weak markov property namely, causal [10], semi causal [8,9] and non causal [1-3]. In contrast with the 1D case where a process obeying an ARMA model is a projection of a vector markov process, the general ARMA model in the 2D case is neither markovian according to any of the three definitions nor a projection of another markov process.

The other topic to be covered is the estimation of parameters in a model to fit a given finite image. The parameter estimates are computed from the empirical correlations by solving linear equations. There are no iterations in contrast with the 1D ARMA model parameter estimation problems.

II. THE ARMA MODEL

Let y(s), $s \in L$ be a certain two-dimensional field of random variables, $L = \{(j,k)\text{: } j,k \text{ are integers}\}$. Let $y(\cdot)$ be stationary and have a spectral density $S_y(z)$.

$$S_y(z) = \nu B(z)/A(z),\ z = [\exp 2\pi i\lambda_1, \exp 2\pi i\lambda_2],\ i = \sqrt{-1} \tag{2.1}$$

where $A(z) = 1 - \sum_{r\in N} \theta_r z^r,\ B(z) = 1 + \sum_{r\in N_2} \phi_r z^r, \theta_r = \theta_{-r},\ \phi_r = \phi_{-r}.$ (2.2)

N_i are finite subsets of L and are symmetric, i.e., if $(j,k) \in N_i$ then $(-j,-k) \in N_i$, $i = 1,2$. $N_2 \subseteq N_1$.

Let $A(z) \neq 0$, for all z such that $|z_i| = 1$, $i = 1,2$, (2.3)

$B(z) \neq 0$, for all z such that $|z_i| = 1$, $i = 1,2$, (2.4)

Conditions (2.3) and (2.4) assure that S(z) is finite, and nonzero positive for all real $\lambda = (\lambda_1,\lambda_2)$. Our aim is to develop a difference equation representation for $y(\cdot)$ valid for any spectral density $S(\cdot)$ obeying (2.1)-(2.4). The recursive simultaneous ARMA model [7] or any other recursive model is automatically ruled out for its lack of generality.

Theorem 1: The stationary random field $y(\cdot)$ with its spectral density in (2.1) obeying the conditions (2.2)-(2.4) obeys the following bilateral difference equation (2.5), the bilateral autoregressive moving average model.

$$y(s) = \sum_{r\in N_1} \theta_r y(s+r) + \sqrt{\nu}\, e(s), \tag{2.5}$$

where the correlated input sequence $e(\cdot)$ has zero mean and spectral density B(z)A(z).

$$S_e(z) = B(z)A(z), \tag{2.6}$$

or equivalently the correlation function in (2.7).

$$E[e(t)e(t+s)] = -\sum_{r\in N_1} \phi_r \cdot \theta_{s-r} \quad \text{if} \quad s \in N'',\ \text{and 0 otherwise} \tag{2.7}$$

where $N_i' = N_i \cup \{0,0\}$, $N'' = \{r+s\text{: } r \in N_1', s \in N_2'\}$.

$$\phi_{0,0} = \theta_{0,0} = 1,\ \phi_s = 0, \quad \text{if } s \notin N_2',\ \theta_r = 0 \quad \text{if } r \notin N_1'$$

Proof: If y(•) obeying (2.5) exists, then taking spectral density of both sides of the equation and using (2.6) indicates that the spectral density of y(•) is as in (2.1). Thus all we have to show is the existence of a sequence y(•) obeying (2.5). This will be done by construction.

Let $\{w(s),\ s \in L\}$ be an infinite sequence of independent and identically distributed random variables having mean zero and variance unity. Let $\tilde{w}(\cdot)$, $\tilde{e}(\cdot)$, $\tilde{y}(\cdot)$ stand respectively for the fourier transform of the infinite sequence w(•), e(•) and y(•). Compute $\tilde{e}(z)$.

$$\tilde{e}(z) = \sqrt{B(z)A(z)}\ \tilde{w}(z).$$

Then the fourier inverse of $\tilde{e}(z)$ yields the sequence $\{e(s),\ s \in L\}$ having zero mean and spectral density B(z)A(z). Compute the sequence $\tilde{y}(z) = \sqrt{\nu}\ \tilde{e}(z)/A(z)$.

$$\text{Equivalently, } (1 - \sum \theta_r z^r)\tilde{y}(z) = \sqrt{\nu}\ \tilde{e}(z), \tag{2.9}$$

The fourier inverse of $\tilde{y}(\cdot)$ yields the desired sequence y(•) obeying (2.5) since (2.5) is the result of the inverse fourier operation on (2.9).

Comment 1: The sequence e(•) is correlated with y(•). The corresponding cross correlation can be found from the corresponding cross spectral density

$$S_{ey}(z) = (\sqrt{\nu}/A(z))S_{ee}(z) = \sqrt{\nu}\, B(z), \tag{2.10}$$

$$\text{i.e., } E[e(t)y(t+r)] = \sqrt{\nu}\ \phi_r, \quad \text{if } r \in N_2', \text{ and } 0 \text{ otherwise} \tag{2.11}$$

Comment 2: (Recursive ARMA): Consider the special case when the polynomial A(z) possesses a factorization as indicated below:

$$A(z) = K\, D_1(z)D_1(z^{-1}) \tag{2.12}$$

where $D_1(z) = 1 - \sum_{r \in N_4} d_r z^r$, N_4 is finite, $(0,0) \notin N_4$, $N_4 \subset L^-$. N_4 is a subset of the nonsymmetrical half plane L^- [10] indicated in figure 1.

$$L^- = \{(j,k): \text{If } k < 0, -\infty < j < \infty; \text{ If } k = 0, j < 0\}$$

$$\text{In this case, we can rewrite (2.5) as: } D_1(z)y(s) = \sqrt{\nu}\ \ u(s), \tag{2.13}$$

where u(•) has the spectral density B(z). The equation (2.13) can be rewritten in the *recursive* form

$$y(s) = \sum_{r \in N_4} d_r y(s+r) + \sqrt{\nu}\ u(s), \tag{2.14}$$

This is the form mentioned in [7]. If in addition B(z) has also a factorization similar to (2.12), say

$$B(z) = K_2 D_2(z)D_2(z^{-1}),\ D_2(z) = 1 + \sum_{r \in N_5} d_r' z^r.$$

and N_5 is a finite subset of L^-, excluding (0,0), then we can rewrite (2.14) as

$$y(s) = \sum_{r\in N_4} d_r y(s+r) + \sqrt{\nu''}(w(s) + \sum_{r\in N_5} d'_r w(s+r)), \qquad (2.15)$$

where w(•) is I.I.D. (0,1) sequence. The above equation is the analog of the traditional ARMA model in time series.

If in (2.15), in addition, $B(z) = 1$ or equivalently, $d'_r = 0$, then the corresponding process y(•) is said to possess the weak linear *causal* markov property, i.e.

$$E[y(s) \mid \text{all } y(s+r), r \in L^-] = \sum_{r\in N_4} d_r\, y(s+r),\ N_4 \subset L^-, \qquad (2.16)$$

In this case, it is possible to divide the image at any point s into 3 parts namely s is the present, the set $\{s+r: r \in L^-\}$ is the past and $\{s+r, r \neq (0,0), r \notin L^-\}$ is the future.

Let us evaluate the conditional expectation of y(s) given all other values for the general model in (2.5).

Theorem 2: The sequence y(•) in Th. 1 obeying (2.5) has the following conditional expectation and variance

$$y_1(s) \triangleq E[y(s) \mid \text{all } y(s+r), r \neq (0,0)] = \sum_{r\neq(0,0)} g_r y(s+r), \qquad (2.17)$$

$$E[(y(s) - y_1(s))^2] = \nu/K, \qquad (2.18)$$

where K and g_r are defined as

$$K(1 - \sum_{r\in L'} g_r z^r) = A(z)/B(z),\ L' = L - (0,0),\ g_r = g_{-r} \qquad (2.19)$$

Proof: Let $G(z) = \sum_{r\in L} g_r z^r$, $v(s) \triangleq y(s) - y_1(s) = (1 - G(z))y(s)$, (2.20)

The cross spectral density between v and y is

$$\begin{aligned} S_{vy}(z) &= (1 - G(z))S_{yy}(z) \\ &= \nu/K, \text{ from (2.1).} \end{aligned}$$

Hence, $E[u(s)y(s+r)] = 0 \quad \forall r \neq (0,0)$

Hence (2.17) is true. To prove (2.18),

$$\begin{aligned} S_{vv}(z) &= (1 - G(z))^2 S_{yy}(z), \text{ from (2.20)} \\ &= (\nu/K)(1 - \textstyle\sum g_r z^r), \text{ from (2.1) and (2.19).} \end{aligned}$$

Hence $E[v^2(s)] = \nu/K$.

The conditional expectation in (2.17) has, in general, an infinite number of terms. The next question is the determination of conditions under which the conditional expectation in (2.17) has a finite number of terms. The answer is in Theorem 3.

Definition: A sequence y(•) is weak *noncausal* markov if the following is true:

$$E[y(s) \mid \text{all } y(s+r), r \neq 0,0] = E[y(s) \mid \text{all } y(s+r), r \in N, N \text{ is finite and symmetric}, (0,0) \notin N]$$

Theorem 3: A stationary sequence is weak non causal markov and possesses a finite linear conditional expectation indicated in (2.21) if and only if the process y(•) has an all pole spectral density, i.e., B(z) in (2.1) is a constant

$$E[y(s) \mid \text{all } y(s+r), r \neq (0,0)] = \sum_{r \in N_1} g_r y(s+r), \tag{2.21}$$

where $(0,0) \notin N_1$ and N_1 is symmetric and finite, $g_r = g_{-r}$.

Comment 1: Parts of the theorem 3 have been known in the literature [1-3]. The aim of giving the theorem is to show the equivalence of the following three statements.

(i) y(•) has the conditional expectation in (2.21).
(ii) y(•) has an all pole spectral density $\nu/A(z)$.
(iii) y(•) obeys the *conditional AR* model in (2.5) where the driving input e(•) has the spectral density A(z).

This equivalence is never explicitly stated in the literature. For instance in [2] both (i) and (iii) are together used in defining the CAR model.

Comment 2: Every sequence y(•) which is causal markov and has the linear expectation in (2.16) defined by a neighbor set N_4 also possesses the non causal markov property in (2.21) with neighbor set N_1, $N_1 \supset N_4$. The reverse is not true [3].

III. PARAMETER ESTIMATION FROM CORRELATIONS

Given a finite image over the MxM grid, we want to develop a procedure for fitting an ARMA model to it, i.e., estimating the unknown parameters in it after fixing the neighbor sets N_1 and N_2. The parameters are estimated using the estimated correlations and it is independent of the density of the image. The method is computationally easy, and does not involve any iteration. This state of affairs is in contrast to the parameter estimation of ARMA models in the time series case. Note that here $\theta_r = \theta(r)$ and $R_r = R(r)$.

Let the spectral density of the process y(•) be $\nu B(z)/A(z)$.

$$A(z) = 1 - \sum_{r \in N_{S1}} \theta_r(z^r + z^{-r}),\ B(z) = 1 + \sum_{r \in N_{S2}} \phi_r(z^r + z^{-r}).$$

N_{S1} and N_{S2} are anti-symmetric subsets of $\{L - \underset{\sim}{0}\}$ i.e., if $s \in N_{S1}$ then $-s \notin N_{S1}$. Similarly for N_{S2}. $N_{S2} \subseteq N_{S1}$.

Let $N_{S1} = \{r_1, \ldots, r_{m_1}\}$, $N_{S2} = \{r_1, \ldots, r_{m_2}\}$, $\#N_{S1} = m_1$, $\#N_{S2} = m_2$. The corresponding ARMA model equation is

$$y(s) = \sum_r \theta_r(y(s+r) + y(s-r)) + \sqrt{\nu}\, e(s), \tag{3.1}$$

where $e(\cdot)$ has the cross spectral density: $S_{ey}(z) = \sqrt{\nu}\,B(z)$.

i.e., $E[e(s)y(s+t)] = \sqrt{\nu}$, if $t = \underset{\sim}{0}$

$$= \sqrt{\nu}\,\phi_r, \text{ if } r \in N_{S2} \text{ or } -r \in N_{S2} \tag{3.2}$$

$$= 0, \text{ otherwise}$$

If $m_2 > 0$ choose an antisymmetric set N_{S3} having m_2 members such that

(i) If $r \in N_{S3}$, then $r \notin N_{S1}$ or $-r \notin N_{S1}$ and $-r \notin N_{S3}$, (ii) $(0,0) \notin N_{S3}$, (iii) $N_3 = \{s\colon s \in N_{S1} \text{ or } -s \in N_{S1} \text{ or } s \in N_{S3} \text{ or } -s \in N_{S3}\}$. The members of N_3 constitute $2(m_1 + m_2)$ nearest neighbors of $(0,0)$. N_3 need not be unique.

Let $N_{S3} = \{t_1,\ldots,t_{m_2}\}$. $N_{S4} \triangleq N_{S1} - N_{S2} = t_{m_2+1}, t_{m_2+2},\ldots,t_{m_1}$ i.e., $r_j = t_j$, $j = m_2+1,\ldots,m_1$. We will obtain an explicit expression for the coefficients θ_r, ϕ_r, ν in terms of the correlations $R(s)$.

Multiply (3.1) by $y(\underset{\sim}{0})$ and take expectation using (3.2).

$$R(0) = 2\sum_{j=1}^{m_1} \theta(r)R(r_j) + \nu, \tag{3.3}$$

Multiply (3.1) by $y(s+r_i)$, $i=1,\ldots,m_2$, and take expectation using (3.2)

$$R(r_i) = \sum_{j=1}^{m_1} \theta(r_j)[R(r_i+r_j) + R(r_i-r_j)] + \nu\phi(r_i),\ i=1,\ldots,m_2, \tag{3.4}$$

Multiply (3.1) by $y(s+t_i)$, $i=1,\ldots,m_1$ and take expectation using (3.2)

$$R(t_i) = \sum_{j=1}^{m_1} \theta(r_j)\{R(t_i+r_j) + R(t_i-r_j)\},\ i=1,\ldots,m_1, \tag{3.5}$$

Solve the m_1 equations in (3.5) for $\theta(r_1), \ldots, \theta(r_{m_1})$.

$$\underset{\sim}{\theta} = \text{column } [\theta(r_1), \ldots, \theta(r_{m_1})] = \underset{\sim}{F}^{-1}\underset{\sim}{R}, \text{ where} \tag{3.6}$$

$\underset{\sim}{R} = \text{Col. } [R(t_1),\ldots,R(t_{m_1})]$. $\underset{\sim}{F} = \{f_{ij},\ i,j = 1,\ldots,m_1\}$, $f_{ij} = R(t_i+r_j) + R(t_i-r_j)$,

Next solve (3.3) for ν.

$$\nu = R(0) - 2\,\underset{\sim}{\theta}^T\underset{\sim}{H},\ \underset{\sim}{H} = \text{col. } [R(r_1),\ldots,R(r_{m_1})] \tag{3.7}$$

Finally solve (3.4) for $\underset{\sim}{\phi}$:

$$\underset{\sim}{\phi} = \text{Col.}[\phi(r_1), \ldots, \phi(r_{m_2})],\ G = \{g_{ij},\ i=1,\ldots,m_2;\ j=1,\ldots,m_1\}$$

$$K = \text{Col.}[R(r_1),\ldots,R(r_{m_2})],\ g_{ij} = R(r_i+r_j) + R(r_i-r_j),\ \underset{\sim}{\phi} = [1/\nu][K - \underset{\sim}{G}\underset{\sim}{0}], \tag{3.8}$$

Thus the 4 steps are

(i) Choose the set N_{S3}.

(ii) Estimate the various correlation needed in R, H, F, G, K.

$$\hat{R}(r) = \frac{1}{M_1 - 1} \sum_s y(s)y(s+r)$$

where summation extends over all valid s in Ω and M_1 is the number of admissable values of s. Let the estimated quantities be $\hat{R}$, $\hat{H}$, $\hat{F}$, $\hat{G}$.

(iii) The estimates of $\hat{\theta}$,$\hat{\nu}$ and $\hat{\phi}$ are:

$$\hat{\theta} = (\underset{\sim}{\hat{F}})^{-1}\hat{R},\ \hat{\nu} = \hat{R}(\underset{\sim}{0}) - 2(\underset{\sim}{\hat{0}})^T\hat{H},\ \hat{\phi} = (1/\hat{\nu})[\hat{K} - \hat{G}\hat{\theta}],$$

It is easy to show that the estimates are consistent, i.e., as the size of the image M goes to infinity, the estimates tend to their true values provided A(z) and B(z) do not have any common factors.

Comment 3: The estimation procedure would work even if N_{S3} is chosen arbitrarily as long as it does not intersect N_{S1}, i.e., conditions (i) and (ii) only are satisfied. However, among all such possibilities, the suggested N_{S3} yields the estimates with the highest accuracy estimates. The reason is that the error in the estimate $\hat{R}(r)$ is directly proportional to $\|r\|$.

Example 1: $N_{S1} = \{(1,0),(0,1)\}$, $N_{S2} = \{(1,0)\}$, $m_1 = 2$, $m_2 = 1$, $m_3 = m_1 - m_2 = 1$.

Our choice of N_{S3} is: $N_{S3} = \{(1,1)\}$, $r_1 = (1,0)$, $t_1 = (1,1)$, $r_2 = t_2 = (0,1)$.

$$\underset{\sim}{F} = \begin{pmatrix} R_{2,1} + R_{0,1}, R_{1,2} + R_{1,0} \\ R_{1,1} + R_{1,-1}, R_{0,2} + R_{0,0} \end{pmatrix},\quad \underset{\sim}{R} = \text{Col.}\ (R_{1,1}, R_{0,1}),\ \underset{\sim}{H} = \text{Col.}\ (R_{1,0}, R_{0,1})$$

$$\underset{\sim}{\theta} = \underset{\sim}{F}^{-1}\underset{\sim}{R},\ \nu = R_{0,0} - (\underset{\sim}{\theta})^T\underset{\sim}{H},\ K = R_{1,0}.$$

$$\underset{\sim}{G} = [R_{2,0} + R_{0,0}, R_{1,1} + R_{1,-1}],\ \phi = (1/\nu)[K - G\theta]$$

REFERENCES

1. Rosanov, Y. A., "On Gaussian Fields with Given Conditional Distributions," Theory of Prob. and its Applications, Vol. XII, #3, 1967, pp. 381-391.
2. Woods, J. W., "Two-Dimensional Discrete Markov Random Fields," *IEEE Trans. on Inform. Theory,* Vol. IT-18, pp. 232-249, March 1972.
3. Besag, J. E., "Spatial Interaction and Statistical Analysis of Lattice Systems," *J. Royal Stat. Soc.,* Ser. B., Vol. B-36, pp. 192-236, 1974.
4. Chellappa, R. and Kashyap, R. L., "Digital Image Restoration Using Spatial Interaction Models," *IEEE Trans. on Acoust., Speech and Signal Proc.,* pp. 461-472, June 1982.

5. McCormick, B. H. and Jayaramanurthy, S. N., "Time Series Models for Texture Synthesis," Int. J. Comput. Inform. Sci., Vol. 4, pp. 329-343, December 1974.
6. Delp, E. J. and Kashyap, R. L. and Mitchell, O. R., "Image Data Compression Using Autoregressive Time Series Models," Pattern Recognition, Vol. 11, 1979, pp. 313-323.
7. Strintzis, M. G., "Dynamic Representation and Recursive Estimation of Cyclic and Two-Dimensional Processes," *IEEE Trans. Automatic Control,* Vol. AC-23, # 5, 1978, pp. 801-808.
8. Jain, A. K. and Jain, J. R., "Partial Difference Equations and Finite Difference Methods in Image Processing Part 2: Image Restoration," *IEEE Trans. Automat. Contrl.,* Vol. AC-23, pp. 817-833, October 1978.
9. Jain, A. K., "Advances in Mathematical Models for Image Processing," Proc. IEEE, Vol. 69, pp. 512-528, May 1981.
10. Whittle, P., "On the Stationary Processes in the Plane," Biometrika, Vol. 41, pp. 434-449, 1954.
11. Kashyap, R. L., "Random Field Models on Finite Lattices for Finite Images," Proceedings of the Conference on Information Sciences, Johns Hopkins University, May 1981.
12. Kashyap, R. L. and Chellappa, R., "Estimation and Choice of Neighbors in Spatial Interaction Models of Images," *IEEE Trans. on Information Theory,* January 1983.
13. Cross, G. R. and Jain, A. K., "Markov Random Field Texture Models," Proc. of IEEE Computer Society Conf. on Pattern Recognition and Image Processing, Dallas, TX, pp. 597-602, August 1981.
14. Brillinger, D. R., *Time Series: Data Analysis and Theory,* Holt, Rinehart & Winston, 1975.
15. Chellappa, R. and Kung, S. Y., "On Two-Dimensional Markov Spectral Estimation," Proceedings of 1982 CISS, Princeton, March 1982.

STOCHASTIC CHARACTERIZATION OF OSCILLATOR PHASE INSTABILITIES

VINCENT C. VANNICOLA
Rome Air Development Center, Griffiss Air Force Base, NY 13441

Pramod K. Varshney
Electrical & Computer Engineering Department, Syracuse University, Syracuse, NY 13210

SUMMARY

Precision frequency sources such as quartz-crystal oscillators and atomic frequency standards and clocks find a wide variety of applications in many engineering fields [1], [2]. Some important application areas are Doppler radar systems; satellite and spacecrafts for guidance, tracking and communications; range measurements and digital communication systems. In these and other coherent information processing systems, stability of the precision frequency sources is of prime importance in maintaining synchronization. Most practical sources, however, exhibit some degree of frequency instability. This may be due to internal or external environmental conditions which the frequency source is exposed to such as additive and multiplicative noise, temperature fluctuations, supply voltage variations, magnetic field, atmospheric pressure, humidity and vibration. These and other time dependent physical causes generate random time-varying fluctuations and instabilities in the parameters which characterize the oscillator, viz., frequency and phase, which result in a degradation in the performance of the coherent information processing systems. In order to evaluate the actual system performance, an accurate characterization of the instability is essential [3], [4]. By phase-frequency instability we mean the degree to which an oscillator produces random unwanted and time dependent departures from a pure sine wave. In the spectrum representation, it is the degree to which these departures cause the oscillator specular component to disperse. The consequences of instability are losses in temporal synchronization and spectral resolution in the associated system. The objective in this paper is to characterize the undesired time-dependent frequency departures and then utilize it for the evaluation of their effects on the performance of coherent information processing systems.

We shall assume the following oscillator signal model

$$b(u) = e^{jz(u)}.$$

In this oscillator rf signal, the phase random process, z(u), generates the random phase fluctuations. We discuss the modeling of this phase random process, z(u). The model uses white phase, random walk phase and random walk frequency processes as well as drift phase and frequency terms. First, the phase random process is expressed as a point vector and it's covariance matrix is determined. The autocorrelation function is determined by taking the characteristic function of the phase covariance matrix. Through the Fourier Transform, the power spectral density is derived for the oscillator rf signal perturbed by the phase instabilities. Furthermore, a procedure is proposed which involves the covariance matrix of the phase random process in conjunction with the characteristic function for determining higher order moments of the rf signal. Spectral dispersion caused by the oscillator phase instabilities is also considered. Details of this work are available in [5].

REFERENCES

1. "IEEE-NASA Symposium on Short Term Frequency Stability", Goddard Space Flight Center, Greenbelt, Md., Nov. 1964.
2. "Special Issue on Frequency Stability", Proc. IEEE, Vol. 54, pp. 101-338, Feb. 1966.
3. J. A. Barnes, et al, "Characterization of Frequency Stability", IEEE Trans. Instrum. Meas., Vol. IM-20, pp. 105-120, May 1971.
4. J. Rutman, "Characterization of Phase and Frequency Instabilities in Precision Frequency Sources: Fifteen Years of Progress", Proc. IEEE, Vol. 66, No. 9, pp. 1048-1075, Sept. 1978.
5. V. C. Vannicola, "Modeling and Properties of Modulated RF Signals Perturbed by Oscillator Phase Instabilities and Resulting Spectral Dispersion", PhD. Dissertation, Syracuse University, May 1982.

ERGODICITY TESTS FOR MULTI-DIMENSIONAL MARKOV CHAINS

LINN I. SENNOTT
Department of Mathematics
Illinois State University
Normal, IL 61761

An irreducible aperiodic Markov chain (or continuous time Markov process) is said to be ergodic if it has a steady-state distribution. The stability of a queueing system often reduces to the ergodicity of an underlying multi-dimensional Markov chain or process. The practitioner is then faced with the potentially difficult job of determining the values of parameters in a model that guarantee its ergodicity. In this paper we discuss some tools that can often help us decide whether or not a chain or process on a multi-dimensional state space is ergodic.

In the paper 'Mean drifts and the non-ergodicity of Markov chains' (L. Sennott, P. Humblet, and R. Tweedie), we developed a tool for proving the non-ergodicity of Markov chains with state space 0,1,... The present paper extends this result to multi-dimensional chains and processes.

In the positive direction, methods for proving ergodicity are reviewed. Current results in the literature often assume the chain has uniformly bounded one-step transition probabilities, a property not satisfied by many Markov chains and processes used to model communication and/or computer systems. We give examples showing the applicability of our results to these types of models.

Our results will often allow us to readily decide whether or not a model is ergodic. However, they are not powerful enough to distinguish between transience and null recurrence. This distinction is not of crucial importance in modeling.

EFFECTS OF A ZERO MEMORY NONLINEARITY ON THE SPECTRUM OF A RANDOM PROCESS

GARY L. WISE
Departments of Electrical Engineering and Mathematics
University of Texas at Austin
Austin, Texas 78712

ABSTRACT

This paper examines some extensions of existing results concerning the effects of a zero memory nonlinearity on the spectrum of a random process. A result concerning the effect of a zero memory nonlinearity on the second moment bandwidth of randomly modulated Gaussian processes is presented. Also, a result is presented concerning the p-th absolute moment of the bandwidth of nonlinearly distorted Gaussian processes.

I. INTRODUCTION AND PRELIMINARIES

Spectral properties of stationary random processes are of concern in several aspects of statistical communication theory and other areas of engineering. If a stationary random process is subjected to some form of nonlinear distortion, it would often be of interest to know the effect of this distortion upon the spectral properties of the random process. In statistical communication theory, as well as other areas of engineering, we are frequently interested in zero memory nonlinearities with random inputs. For example, limiters, rectifiers, power law devices, and other zero memory nonlinearities are an integral part of many systems. In this paper we will be concerned with time invariant zero memory nonlinearities. Our attention will be directed toward the characterization of bandwidth properties of the output of a zero memory nonlinearity with a stationary random input.

We will let $\{G(t),\ t \in \mathbb{R}\}$ denote a stationary, mean square continuous Gaussian process with zero mean, a positive variance σ^2, and a nonconstant autocorrelation function $\sigma^2\rho(\tau)$; R will denote a second order random variable, independent of the random process G(t), such that $P(R=0) < 1$; and $\{A(t),\ t \in \mathbb{R}\}$ will denote a second order, mean square continuous, second order stationary random process with a positive second moment, such that the random processes A(t) and G(t) and the random variable R are mutually independent. By second order stationarity we refer to the property that the bivariate distribution function associated with the random process of interest is invariant to a shift of the time axis. The class of all second order, mean square continuous, second order stationary random processes possessing a bounded spectrum will be denoted by B_{spec}. If X(t) is a second order, mean square continuous, second order stationary random process such that $P(X(t) = 0) < 1$, then $BW\{X(t)\}$ will denote the second moment of the normalized spectral distribution of X(t). When X(t) is second order stationary, we will define $\mathscr{G}[X(t)]$ to be the class of all Borel measurable functions $g\colon \mathbb{R} \to \mathbb{R}$ such that g is not almost everywhere constant and $E\{(g[X(t)])^2\} < \infty$. The set of all polynomials and all Borel measurable functions equal almost everywhere to a polynomial will be denoted by $\mathscr{P}$.

Recall that if X(t) is a second order random process that is second order stationary, then mean square continuity is a necessary and sufficient condition for the popular spectral representation of X(t) [1]. We will be interested in spectral properties of g[X(t)], when $g \in \mathscr{G}[X(t)]$. The following result[2] shows that we will not lose the mean square continuity by such a transformation.

Theorem 1: If $X(t)$ is a second order, mean square continuous random process that is second order stationary, and if $g \in \mathscr{G}[X(t)]$, then $g[X(t)]$ is mean square continuous.

The following six theorems summarize existing developments.

Theorem 2 [3]: If $g \in \mathscr{G}[R\ G(t)]$, then $g[R\ G(t)] \in B_{\text{spec}}$ if and only if $G(t) \in B_{\text{spec}}$ and $g \in \mathscr{P}$.

Theorem 3 [3]: If $g \in \mathscr{G}[A(t) + R\ G(t)]$ and $G(t) \notin B_{\text{spec}}$, then $g[A(t) + R\ G(t)] \notin B_{\text{spec}}$.

Theorem 4 [3]: If $g \in \mathscr{G}[A(t) + R\ G(t)]$ and $g \notin \mathscr{P}$, then $g[A(t) + R\ G(t)] \notin B_{\text{spec}}$.

Theorem 5 [4]: If $g \in \mathscr{G}[A(t)G(t)]$ and $G(t) \notin B_{\text{spec}}$, then $g[A(t)G(t)] \notin B_{\text{spec}}$.

Theorem 6 [4]: If $g \in \mathscr{G}[A(t)G(t)]$ and $g \notin \mathscr{P}$, then $g[A(t)G(t)] \notin B_{\text{spec}}$.

Theorem 7 [5]: If $g \in \mathscr{G}[R\ G(t)]$ and $G(t)$ possesses a spectral density function and $E\{g[R\ G(t)]\} = 0$, then $BW\{R\ G(t)\} \leq BW\{g[R\ G(t)]\}$ with equality if and only if g is almost everywhere linear.

Theorems 2-6 are thus concerned with bandlimitedness, while Theorem 7 is concerned with the second moment of the spectral distribution. In the next section we will extend Theorem 7 to a more general class of random processes. Then in the following section we will remove the assumption of the existence of a spectral density from Theorem 7, and we will also consider other moments of the spectral distribution.

II. ANALYSIS

Recall [6] that if U and V are mutually Gaussian random variables with zero means, variances $\sigma^2 > 0$, and correlation coefficient α, if g_1 and g_2 are Borel measurable functions such that $E\{[g_i(U)]^2\} < \infty$, $i = 1,2$, then

$$E\{g_1(U)g_2(V)\} = \sum_{n=0}^{\infty} \alpha^n\, b_n^{(1)} b_n^{(2)} \quad ,$$

where

$$b_n^{(i)} = E\{g_i(U)\, H_n(U/\sigma) \,/\, \sqrt{n!}\,\},$$

where

$$H_n(x) = (-1)^n \exp(x^2/2)\, \frac{d^n}{dX^n} \exp(-x^2/2),$$

and where the convergence is absolute and uniform in α. Let

$X(t) = R\,A(t)G(t)$ and let $g(\cdot)$ be an odd function belonging to $\mathscr{G}[X(t)]$. Using the preceding result we have that the following conditional expectation is almost surely given by

$$E\{g[X(t)]\,g[X(s)] \mid R, A(t), A(s)\} =$$

$$\sum_{n=1}^{\infty} [\rho(t-s)]^n \; b_n[R\,A(t)]\,b_n[R\,A(s)],$$

where

$$b_n(y) = E\{g[y\,G(t)]\,H_n(G(t)/\sigma)\,/\,\sqrt{n!}\,\}.$$

Thus we see that the output autocorrelation function is given by

$$E\{g[X(t)]\,g[X(s)]\} = E\left\{\sum_{n=0}^{\infty} [\rho(t-s)]^n\, b_n[R\,A(t)]\,b_n[A(s)]\right\}$$

$$= \sum_{n=0}^{\infty} [\rho(t-s)]^n\, E\{b_n[R\,A(t)]\,b_n[R\,A(s)]\},$$

where the dominated convergence theorem is used to exchange the expectation and infinite summation, i.e.

$$|[\rho(t-s)]^n\, b_n[R\,A(t)]\,b_n[R\,A(s)]| \le (b_n[R\,A(t)])^2 + (b_n[R\,A(s)])^2$$

and

$$E\left\{\sum_{n=0}^{\infty} (b_n[R\,A(t)]^2\right\} = E\{g[X(t)])^2\} < \infty.$$

Notice that $b_n[R\,A(t)]$ is a wide sense stationary random process. Thus $E\{b_n[R\,A(t)]\,b_n[R\,A(s)]\}$ is a nonnegative definite function of $(t-s)$, say $R_n(t-s)$, and it follows from Theorem 1 that $R_n(\cdot)$ is continuous. Notice that $R_0(\tau) \equiv 0$, since g is odd.

Thus

$$E\{g[X(t)]\,g[X(s)]\} = \sum_{n=1}^{\infty} [\rho(t-s)]^n\, R_n(t-s).$$

Letting $r(\tau)$ denote the autocorrelation function of $g[X(t)]$, we have

$$r(\tau) = \sum_{n=1}^{\infty} [\rho(\tau)]^n\, R_n(\tau). \tag{1}$$

Since $\quad r(0) = \sum_{n=1}^{\infty} R_n(0) = E\{(g[X(t)]^2\} < \infty ,$

we see that the convergence in (1) is uniform in τ.

We define the spectral distribution functions corresponding to r, ρ, and R_n as F, D, and D_n, respectively. (All spectral distribution functions are taken to be zero at $-\infty$ and right continuous.) Since (1) converges uniformly, we see that

$$F(\omega) = D(\omega) * D_1(\omega) + \sum_{n=2}^{\infty} \left([D(\omega)]^{*n}\right) * D_n(\omega) ,$$

where the asterisk denotes convolution scaled by $1/2\pi$ and $[H(\cdot)]^{*n}$ denotes an (n-1)-fold scaled convolution of $H(\cdot)$ with itself. It follows from the monotone convergence theorem that

$$\int \omega^2 dF(\omega) = \sum_{n=1}^{\infty} \int \omega^2 dG_n(\omega)$$

where $G_1(\omega) = D(\omega) * D_1(\omega)$ and for $n \geq 2$,

$$G_n(\omega) = \left([D(\omega)]^{*n}\right) * D_n(\omega).$$

Hence, if $\int \omega^2 dG_n(\omega) < \infty$ for all $n \geq 1$, we see [7] that

$$\int \omega^2 dF(\omega) = \sum_{n=1}^{\infty} - \frac{d^2}{d\tau^2} \left([\rho(\tau)]^n R_n(\tau)\right) \Big|_{\tau=0}$$

On the other hand, if for some $n \geq 1$ $\int \omega^2 dG_n(\omega) = \infty$, then $\int \omega^2 dF(\omega) = \infty$. Thus $\int \omega^2 dF(\omega) < \infty$ implies that $[\rho(\tau)]^n R_n(\tau)$ is twice-differentiable for all n. Notice that if a product of continuous nonnegative definite functions is twice-differentiable, then each of the continuous nonnegative definite functions is twice-differentiable, i.e. appeal to the analogy of characteristic functions and the sum of independent second order random variables. Thus if $\int \omega^2 dF(\omega) < \infty$, then (noting that $F(\infty) > 0$)

$$\frac{\int \omega^2 dF(\omega)}{F(\infty)} = \frac{-1}{F(\infty)} \sum_{n=1}^{\infty} R_n''(0) - \frac{\rho''(0)}{F(\infty)} \sum_{n=1}^{\infty} n R_n(0) \quad ;$$

or, equivalently, noticing that $F(\infty) = \sum_{n=1}^{\infty} R_n(0)$,

$$\frac{\int \omega^2 dF(\omega)}{-\rho''(o) F(\infty)} = \frac{\sum_{n=1}^{\infty} R_n''(0)}{\rho''(0) F(\infty)} + \frac{\sum_{n=1}^{\infty} nR_n(0)}{\sum_{n=1}^{\infty} R_n(0)} \geq 1, \tag{2}$$

where we observe that $\rho''(0) < 0$. That is, if $\rho''(0) = 0$, then

$\int \omega^2 dD(\omega) = 0$, which from Bochner's theorem would imply that $\rho(\tau)$ is constant. Thus the first summand in the middle of (2) is nonnegative.

Notice that if the random process $A(\tau)$ is identically one, then $R_n''(0) = 0$ for $n \geq 1$, and in this case we have

$$\frac{\int \omega^2 dF(\omega)}{-\rho''(0)F(\infty)} = \frac{\sum_{n=1}^{\infty} nR_n(0)}{\sum_{n=1}^{\infty} R_n(0)} \geq 1. \tag{3}$$

In this situation equality prevails if and only if $R_n(0) = 0$ for $n \geq 2$. This is equivalent to $b_n(R) = 0$ with probability one for $n \geq 2$. Now if g is not equivalent (under Lebesgue measure) to a linear function, then for any $y \neq 0$, $\sum_{n=2}^{\infty} [b_n(y)]^2 > 0$. Thus $P\left(\sum_{n=2}^{\infty} [B_n(R)]^2 > 0\right) = P(R \neq 0) > 0$, which implies that

$$\sum_{n=2}^{\infty} R_n(0) = E\left\{\sum_{n=2}^{\infty} [b_n(R)]^2\right\} > 0 ,$$

that is, the equality follows from the monotone convergence theorem and the inequality follows from a standard integration theorem [8]. Taking the contra-positive, we conclude that equality prevails in (3) if and only if g is almost everywhere linear.

Summarizing these observations, we have the following result.

Theorem 8: If $g \in \mathscr{G}[A(t)R\,G(t)] \cap \mathscr{G}[R\,G(t)]$ and if g is odd, then

$$BW\{RG(t)\} \leq BW\{g[RG(t)]\} \leq BW\{g[A(t)R\,G(t)]\},$$

where the first inequality is met with equality if and only if g is almost everywhere linear.

The above theorem thus pertains to randomly modulated Gaussian processes and yields a considerable amount of insight concerning the spreading effect on the bandwidth caused by the random modulation; for example, $BW\{G(t)\} \leq BW\{A(t)G(t)\}$, where $A(t)$ is any random process having the mild restrictions given earlier.

III. GAUSSIAN INPUTS

Now consider the Gaussian process $G(t)$. Let $g \in \mathscr{G}[G(t)]$, and assume that g is such that $E\{g[G(t)]\} = 0$. Mimicking the development at the beginning of the previous section, we have that $r(\tau) = E\{g[G(t)]\, g[G(t+\tau)]\}$ is given by

$$r(\tau) = \sum_{n=1}^{\infty} b_n^2 \,[\rho(\tau)]^n , \tag{4}$$

where $b_n = E\{g[G(t)]\, H_n(G(t)/\sigma) \,/\, \sqrt{n!}\,\}$

and where the convergence in (4) is uniform in τ. Let

$$p_n = b_n^2 \Big/ \sum_{i=1}^{\infty} b_i^2. \tag{5}$$

Then

$$\frac{r(\tau)}{r(0)} = \sum_{n=1}^{\infty} p_n[\rho(\tau)]^n$$

and

$$F(\omega) \Big/ F(\infty) = p_1 D(\omega) + \sum_{n=1}^{\infty} p_n [D(\omega)]^{*n} ,$$

where F and D, respectively, are the spectral distribution functions associated with r and ρ. Notice that $F(\cdot)/F(\infty)$ can be interpreted as the probability distribution function of the random variable $\sum_{i=1}^{N} Y_i$, where $\{Y_i, Y_2, \ldots\}$ are mutually independent and identically distributed random variables with the probability distribution function $D(\cdot)$, and where N is a random variable, independent from $\{Y_1, Y_2, \ldots\}$, taking values in the positive integers such that $P(N=n) = p_n$. In this context we see that (for $p \geq 2$)

$$\bar{A}_p = \frac{\int |\omega|^p dF(\omega)}{F(\infty)} \tag{6}$$

is given by

$$\bar{A}_p = \sum_{n=0}^{\infty} p_n \quad E\{|Y_1 + \ldots + Y_n|^p\}. \tag{7}$$

Let

$$A_p = \int |\omega|^p \; dD(\omega). \tag{8}$$

Then A_p is given by $E\{|Y_1|^p\}$.

Recalling that $E\{Y_1\} = 0$, we see that

$$\begin{aligned} E\left\{|Y_1 + \ldots + Y_n|^p\right\} &= E\left\{|Y_1 + \ldots + Y_n|^{2(p/2)}\right\} \\ &\geq \left(E\left\{|Y_1 + \ldots + Y_n|^2\right\}\right)^{p/2} \\ &= [n\, E\{(Y_1)^2\}]^{p/2} \\ &= n^{p/2} A_2^{p/2} . \end{aligned}$$

Thus $\overline{A}_p \geq \sum_{n=1}^{\infty} p_n n^{p/2} A_2^{p/2}$. In particular, we see from the above that for $p = 2$, the inequality is met with equality and we have

$\overline{A}_2 = A_2 \sum_{n=1}^{\infty} np_n \geq A_2$. Using this observation for Gaussian processes and and an obvious conditioning argument, we see that we can remove the assumption of the existence of a spectral density in Theorem 7. This is given as the following result.

Theorem 7: If $g \in \mathcal{G}[R\, G(t)]$ and $E\{g[R\, G(t)]\} = 0$, then $BW\{R\, G(t)\} \leq BW\{g[R\, G(t)]\}$, with equality if and only if g is almost everywhere linear.

Returning to (6), we now consider an upper bound on $\overline{A}_p$. Recall from [9] that

$$E\{|Y_1 + \ldots + Y_n|^p\} \leq \frac{n^{p/2}\, 2^{3p/2}}{\sqrt{\pi}} \Gamma\left(\frac{p+1}{2}\right) E\{|Y_1|^p\}.$$

Thus from (7) we see that

$$\overline{A}_p \leq A_p \frac{2^{3p/2}}{\sqrt{\pi}} \Gamma\left(\frac{p+1}{2}\right) \sum_{n=1}^{\infty} p_n n^{p/2} .$$

We summarize these observations as follows.

Theorem 9: Assume that $g \in \mathcal{G}[G(t)]$ and that $E\{g[G(t)]\} = 0$. Let $\overline{A}_p$ and A_p be defined, respectively, by (6) and (8) for $p \geq 2$, and let p_n be given by (5). Then

$$A_2^{p/2} \sum_{n=1}^{\infty} p_n n^{p/2} \leq \overline{A}_p \leq A_p \frac{2^{3p/2}}{\sqrt{\pi}} \Gamma\left(\frac{p+1}{2}\right) \sum_{n=1}^{\infty} p_n n^{p/2} .$$

Acknowledgement

This research was supported by the Air Force Office of Scientific Research under Grant AFOSR-81-0047. The author would also like to acknowledge the encouragement and friendship of the late Dr. Ismail N. Shimi, formerly of the Air Force Office of Scientific Research.

References

[1] M. Loeve, *Probability Theory II*, Springer-Verlag, New York, 1978, pp. 148-149.

[2] G.L. Wise and J.B. Thomas, "On a Completeness Property of Series Expansions of Bivariate Densities," *Journal of Multivariate Analysis*, Vol. 5, pp. 243-247, June 1975.

[3] G.L. Wise, "The Effect of a Zero Memory Nonlinearity on the Bandlimitedness of Contaminated Gaussian Inputs," *IEEE Trans. Information Theory*, Vol. IT-26, pp. 345-347, May, 1980.

[4] G.L. Wise, "Recent Results Concerning the Effects of Nonlinearities on Random Inputs," *Proceedings of the Twelfth Southeastern Symposium on System Theory*, Virginia Beach, Virginia, May 19-20, 1980, pp. 290-294.

[5] G.L. Wise, A.P. Traganitis, and J.B. Thomas, "The Effect of a Memoryless Nonlinearity on the Spectrum of a Random Process," *IEEE Trans. Information Theory*, Vol. IT-23, pp. 84-89, January, 1977.

[6] S. Cambanis and B. Liu, "On the Expansion of a Bivariate Distribution and its Relation to the Output of a Nonlinearity," *IEEE Trans. Information Theory*, Vol. IT-17, pp. 17-25, January, 1971.

[7] J.L. Doob, *Stochastic Processes*, John Wiley, New York, 1953, p. 537.

[8] W. Rudin, *Real and Complex Analysis*, McGraw-Hill, New York, 1974, p. 31.

[9] P. Whittle, "Bounds for the Moments of Linear and Quadratic Forms in Independent Variables," *Theory of Probability and Its Applications*, Vol. 5, pp. 302-305, 1960.

TIME-SPACE TRADEOFFS FOR BACK-TO-BACK FFTs: THE EFFECT OF THE CONNECTION PATTERN*

DAVID A. CARLSON
Department of Electrical and Computer Engineering
University of Massachusetts
Amherst, MA 01003

ABSTRACT

The computation associated with back-to-back applications of a Fast Fourier Transform is composed of two FFT computations, the outputs of one being connected to the inputs of the other according to a specified pattern (one-to-one mapping). In this paper, we examine the effect that the connection pattern has on the time-space tradeoff behavior of such a computation. We model the computation as a directed acyclic graph, and play a well-known pebble game on the graph to obtain the simultaneous time and space requirements of the computation. For the connection pattern between the two FFTs that is defined by the bit reversal permutation (such a connection pattern is associated with an implementation of polynomial multiplication using FFTs), we derive an optimal upper bound on the time required to compute the outputs of such a back-to-back FFT using restricted space. We also present a slightly weaker upper bound on the simultaneous time and space required to compute a back-to-back FFT when the connection pattern is different. We present evidence supporting the conjecture that this weaker upper bound cannot be improved, i.e. that there is a back-to-back FFT having a certain connection pattern for which a matching lower bound can be derived.

INTRODUCTION

The development of lower bounds on the simultaneous resource requirements of algorithmic solutions to a computational problem has received a great deal of interest recently, motivated by a desire to obtain a more complete understanding of a problem's computational complexity. Typical resources that affect the cost of performing a computation are time, measured by the number of machine instructions executed, and space, measured by the number of memory locations referenced. Bounds on the simultaneous time and space requirements of a problem can yield valuable information concerning how these resources are related to each other, and they also complement well knowledge of lower bounds on the individual resources of time and space. For example, there may be a tradeoff between these two quantities, for which the reduction in usage of time causes a corresponding need for an increase in space, and vice-versa. An understanding of the time-space tradeoff characteristics of a given computational problem is of interest, since it can indicate the values of these quantities that can or cannot be achieved simultaneously by an algorithmic solution to the problem.

The simultaneous time and space requirements of a straight-line algorithm can be analyzed by playing the well known pebble game on a directed acyclic graph whose vertices represent operations performed in the algorithm and whose edges denote argument assignments for these operations. Playing the pebble game on such a graph models the allocation of temporary storage locations to hold the values of intermediate computations during the execution of the algorithm. In the game, pebbles are moved from input vertices to output vertices according to three rules: 1) a pebble can be

---

*Work supported in part by NSF grant ECS-04894

placed on an input vertex at any time, 2) a pebble can be removed from any vertex at any time, and 3) a pebble can be placed on a noninput vertex only if all its predecessors have pebbles on them. Time T is measured by the total number of moves (applications of rules 1 and 3) required to pebble all output vertices of the graph, and space S corresponds to the maximum number of pebbles active on the graph at any one time.

When the space available to pebble a graph (perform a computation) is limited, it may be necessary to repeat the pebbling of many non-output vertices, from which a tradeoff between time and space can be observed. A large amount of research concerning the pebble game has dealt with the possible exchange between time and space. Paterson and Hewitt [4] introduced the pebble game, and used it as a technique to study the effects of restricted space on the computation of a linear recursive program. A number of authors [3,5,9] have analyzed graph families corresponding to straight-line algorithms for which a decrease in space below a certain known value causes the time required to pebble a member of the family to grow faster than any polynomial in the size of the graph.

Savage and Swamy [6] were the first to use the pebble game to analyze the simultaneous time and space requirements of a natural algorithm. For an n-input Fast Fourier Transform (FFT), they showed that $T \cdot S = \Theta(n^2)$ by deriving matching (to within a constant factor) upper and lower bounds on the product of time and space. The lower bound is a consequence of the fact that pebbling a set of outputs of the FFT graph of size proportional to the number of pebbles used requires pebbling a large number of inputs. Since there are $\Omega(n/S)$ such sets of outputs and each requires repebbling $\Omega(n)$ inputs, it follows that $T = \Omega(n^2/S)$. Tompa [7] extended this work to obtain similar lower bounds on the time-space product for natural problems such as the computation of a Discrete Fourier Transform. His results rely on the fact that for any straight-line solution to a problem such as the DFT, the corresponding graph has certain connectivity properties between inputs and outputs. By an argument similar to that mentioned above, these connectivity properties imply that a large number of inputs must be repebbled in order to pebble each output set of size proportional to the available space.

The proof method outlined above can also be used to analyze the time-space tradeoff behavior of a graph composed of a sequence of subgraphs (outputs of one subgraph in the sequence are inputs to the next subgraph) as long as each subgraph possesses the proper connectivity property. Both Paul and Tarjan [5] and Tompa [9] construct graph families in this manner for which a decrease in space causes time to increase from polynomial to superpolynomial.

In this paper, we are concerned with the graph composed of a pair of FFT graphs, connected "back-to-back". The computation of "back-to-back" FFTs has applications as a fast algorithm for polynomial multiplication and in signal processing computations. As noted in Tompa [8], a lower bound of $T = \Omega(n^3/S^2)$ for a back-to-back FFT graph on n inputs is an immediate consequence of the lower bound techniques discussed above, and Carlson [2] has developed an alternative technique that yields a lower bound of the form $T = \Omega(n^2 \log n/S)$. Both of these lower bounds are independent of the pattern by which outputs of one FFT graph are connected to inputs of the other.

Here, we consider the effect that the connection pattern between the two FFT subgraphs has on the time-space tradeoff characteristics of the entire graph. Our results indicate that the way in which the two subgraphs are connected can exert a non-trivial influence on the simultaneous time and space requirements of a back-to-back FFT computation, and thus, such behavior is possible for general computations composed in this manner. Specifically, we exhibit a connection pattern for which the lower bound $T=\Omega(n^3/S^2 + n^2\log n/S)$ can be achieved over a large range of space. Since this connection pattern is associated with an implementation of polynomial multiplication using back-to-back Fast Fourier Transforms, the upper bound implies efficient algorithms for polynomial multiplication that are optimal with respect to all "FFT-based" algorithms.

We also derive a somewhat weaker upper bound of the form $T=O(n^3\log(\log n - \log S)/S^2 + n^2\log n/S)$ that applies to a back-to-back FFT computation having a different connection pattern (i.e. for the connection pattern, the above upper bound can be achieved). This upper bound is an asymptotic improvement over previously stated upper bounds on the simultaneous time and space requirements of a back-to-back FFT [7,8]. We present evidence supporting the fact that this upper bound cannot be improved; in other words, that this connection pattern has an associated lower bound that matches the upper bound given above. Such a lower bound would be a loglog n factor increase over $T=\Omega(n^3/S^2 + n^2\log n/S)$ when space $S=O(n^{\varepsilon})$, $0<\varepsilon<1$.

The results outlined above indicate that for a computation composed of two or more subcomputations, it may not be enough to know the time-space tradeoff characteristics of the subcomputations in order to determine the time and space requirements of the entire computation, since the way in which inputs of one subcomputation depend on outputs of another may be a non-negligible influence. Also, the back-to-back FFT computation has practical applications, and the upper bounds presented here address this issue by indicating a spectrum of multi-resource efficient algorithms.

2. Preliminaries

In this section, we give definitions of the FFT and the back-to-back FFT graphs. We also state basic properties of these graphs that will be useful in later analysis of their time-space tradeoff characteristics. The material presented here is similar to that contained in Savage and Swamy [6].

To form the Fourier Transform of a sequence $a=(a_0,\ldots,a_{n-1})$ over a ring R, we must calculate the terms f_i in the sequence $f(a) = (f_0,\ldots f_{n-1})$. The function f_i is defined to be the value of the polynomial

$$p(\chi) = a_0 + a_1\chi + \ldots + a_{n-1}\chi^{n-1}$$

at v^i, v being a principal nth root of unity. The Fast Fourier Transform (FFT) algorithm is derived by splitting the polynomial $p(\chi)$ into its odd and even terms. Recursively calculating the Fourier Transforms of the odd and even terms and combining the results yields the Fourier Transform of the original sequence. Converting this recursive procedure into an iterative one, we obtain a straight-line algorithm, since the operations are executed in a data independent manner. This algorithm can be represented with a graph, which is defined as follows.

Definition: Let $n = 2^d$, $d > 0$. The directed graph FFT[d] = (V[d],E[d], I[d],O[d]) on n inputs has vertex set V[d], edge set E[d], input set I[d] and output vertex set O[d]. It is defined recursively in terms of two disjoint graphs

$$FFT_0[d-1] = (V_0[d-1], E_0[d-1], I_0[d-1], O_0[d-1])$$

and

$$FFT_1[d-1] = (V_1[d-1], E_1[d-1], I_1[d-1], O_1[d-1])$$

on n/2 inputs each as follows. Let

$$I[d] = \{x_0^d, \ldots, x_{2^d-1}^d\}$$

be a set of vertices disjoint from $FFT_0[d-1]$ and $FFT_1[d-1]$. Then

$$O[d] = O_0[d-1] \cup O_1[d-1]$$

$$V[d] = V_0[d-1] \cup V_1[d-1] \cup I[d]$$

$$E[d] = E_0[d-1] \cup E_1[d-1] \cup \Delta[d]$$

where

$$\Delta[d] = \bigcup_{i=0}^{2^{d-1}-1} \delta_i[d]$$

$$\delta_i[d] = ((x_i^d, x_{i,o}^{d-1}), (x_{i+2^{d-1}}^d, x_{i,o}^{d-1}), (x_i^d, x_{i,1}^{d-1}), (x_{i+2^{d-1}}^d, x_{i,1}^{d-1}))$$

Here, (v,w) denotes an edge directed from v to w, and $FFT[0] = \{V[0], \phi, V[0], V[0]\}$ where $V[0] = \{x_0^0\}$ and ϕ is the empty set.

Figure 1 illustrates an FFT graph on 4 inputs and also shows the correspondence that vertices in the graph have to intermediate computations of the FFT algorithm. Note that this graph is essentially equivalent to the definition in Savage and Swamy [6], with input vertices no longer being "scrambled" by the bit reversal permutation.

Lemma 1: For $d \geq 2$ and each $1 \leq j \leq d-1$, the FFT graph FFT[d] can be represented as the composition of 2^{d-j} disjoint FFT graphs on 2^j inputs $\{A_\ell[j] \mid 0 \leq \ell \leq 2^{d-j}-1\}$ with 2^j disjoint FFT graphs on 2^{d-j} inputs $\{B_m[d-j] \mid 0 \leq m < 2^j-1\}$ where the ℓth output of $B_m[d-j]$ is the mth input to $A_\ell[j]$ for $0 \leq \ell \leq 2^{d-j}-1$ and $0 \leq m \leq 2^j-1$. The ith input to FFT[d] is the qth input to $B_r[d-j]$ for the unique integers $q, r \geq 0$ such that $i = q \cdot 2^j + r$ and $r < 2^j$, and the kth output of FFT[d] is the sth output of $A_t[j]$ for the unique integers $s, t \geq 0$ such that $K = t \cdot 2^j + S$ and $t < 2^{d-j}$.

Lemma 2: In FFT[d], the paths from any given input to outputs form a complete binary tree with 2^d leaves (outputs) as do the paths from inputs to any given output.

We now define a back-to-back FFT graph as follows:

Definition: A back-to-back FFT graph on $n=2^d$ inputs, $BBFFT[d,\pi]$, is composed of two disjoint FFT graphs $FFT_0[d]$ and $FFT_1[d]$, each on $n=2^d$ inputs where output $i (o < i \leq 2^d-1)$ of $FFT_0[d]$ is connected to input $\pi(i)$ of $FFT_1[d]$ for some permutation π.

Figure 2 illustrates a back-to-back FFT graph on n=8 inputs where the connection between the two FFT graphs is defined by the identity permutation.

3. Pebbling the Outputs of an FFT Graph in a Certain Order

The basic idea supporting the existence of an improved lower bound for a back-to-back FFT computation is that for each output set of size proportional to the available space, a large number of distinct binary trees contained in the FFT subgraph must be pebbled (this implies pebbling a large set of inputs to the FFT subgraph and is explained in Savage and Swamy [6]). However, there is not enough space available to pebble these binary trees in a synchronized fashion, so that an ordering on the time when inputs to the trees must be pebbled is defined. Dependent on the connection pattern, this ordering defines another ordering for pebbling the outputs of the other FFT subgraph. Thus, in this section, we study the effect that pebbling the outputs of a single FFT graph in a certain order has on the simultaneous time and space requirements of the FFT graph. Again, the effect is a non-trivial one, as evidenced by the development of a lower bound for an ordering in time based on the bit reversal permutation that is significantly stronger than $T=\Omega(n^2/S)$ [6]. The techniques developed in the analysis of the single FFT graph apply in a similar but weaker form to the back-to-back FFT graph with a certain connection pattern, which indicates the existence of the desired lower bound.

In Savage and Swamy [6], a pebbling strategy for the n-input FFT graph is outlined that uses space S and (optimal) time $T=O(n^2/S)$. For the graph FFT [d], when $S \approx 2^j (o<j<d)$, this strategy operates by pebbling trees in the subgraphs $\{B_m[d-j]\}$ whose outputs are inputs to a subgraph $A_{\ell_o}[j]$. Then, $A_{\ell_o}[j]$ is pebbled using a "level-by-level" strategy. The subgraphs $\{A_\ell[j]\}$ can be pebbled in any desired order, but all outputs of an individual subgraph $A_{\ell_o}[j]$ are pebbled before pebbling outputs of another subgraph $A_{\ell_1}[j]$.

The above strategy can be seen to be inefficient if it is required that a single output in $A_{\ell_o}[j]$ be pebbled followed by a single output in $A_{\ell_1}[j]$, etc.; i.e. that the outputs of the FFT graph be pebbled in an order for which outputs pebbled in the same interval of time occur in different subgraphs. Such an ordering on the outputs of FFT[d] is implied by the bit reversal permutation, as evidenced by the following key lemma.

Lemma 3: Let the outputs of FFT[d] be specified by $\{y_o,\ldots y_{2^{d-1}}\}$ and let $B(\chi)$ denote the bit reversal of the d-digit binary representation of χ. Then, for any k $(o\leq k\leq d)$, for the subset of outputs $\{y_{B(c+i)} | o\leq i<2^k\}$ $(c\leq 2^d-2^k)$, $y_{B(c+i)}$ is contained in a subgraph $A_{\ell_i}[d-k]$ of FFT[d] and $A_{\ell_i}[d-k] = A_{\ell_j}[d-k]$ if and only if $i=j$ (equivalently, $B(c+i) = B(c+j)$).

Proof (by induction on d):

Basis: d=o. Trivial.

Induction: Assume true for the case d-1. Consider the set of outputs $Y=\{y_{B(c)},\ldots,y_{B(c+2^k-1)}\}$. It is easy to see that, due to the bit reversal permutation, the subset of outputs $Y_o=\{y_{B(c+i)} | o\leq i\leq 2^k$ and c+i is even$\}$ is contained in $FFT_o[d-1]$, and $Y_i=\{y_{B(c+i)} | o\leq i<2^k$ and c+i is odd$\}$ is contained in $FFT_1[d-1]$. Furthermore, there exist c_1, c_2 such that $Y_0=\{y_{B'(c_1+i),0} \mid 0\leq i<2^{k-1}\}$, $Y_1 = \{y_{B'(c_2+i),1} \mid 0\leq i<2^{k-1}\}$, where $B'(\chi)$ is the bit reversal of the d-1 digit binary representation of χ. Thus, by induction, each output in $Y_0 (Y_1)$ is contained in a distinct subgraph $A_{\ell_0}[d-1-(k-1)]$ of $FFT_0[d-1]$ ($A_{\ell_1}[d-1-(k-1)]$ of $FFT_1[d-1]$). But, since $FFT_0[d-1]$ and $FFT_1[d-1]$ are disjoint, these subgraphs are simply the set of subgraphs $\{A_\ell[d-k]\}$ of $FFT[d]$. Q.E.D.

We are now in a position to prove the main result of this section.

Theorem 1: Let T be the number of moves required to pebble the outputs of an FFT graph $FFT[d]$ on $n=2^d$ inputs in an order specified by the bit reversal permutation using space S. Then, $T=\Omega(n^2(\log_2 S)/S)$.

Proof: First of all, pebbling the outputs of $FFT[d]$ in an order based on the bit reversal permutation means that there exists a set of times $\{t_i \mid 0\leq i<2^d\}$ such that $t_i\leq t_j$ for $i\leq j$ and at time t_i, output $y_{B(i)}$ holds a pebble (here, a time corresponds to a distinct application of rules 1 or 3 of the pebble game). Assume that we have $S\leq 2^{j-1}$ pebbles and consider pebbling a set of 2^j outputs $Y = \{y_{B(c)},\cdots, y_{B(c+2^j-1)}\}$ that must be pebbled in consecutive order. By Lemmas 2 and 3, these outputs appear in distinct subgraphs $\{A_\ell[d-j]\}$, and each is the root of a complete binary tree with 2^{d-j} leaves (inputs to the tree) contained in $A_\ell[d-j]$. At the start of the pebbling process for this set of outputs, at most S of these binary trees have pebbles, so at least $2^j-S\geq 2^{j-1}$ trees must be pebbled in their entirety in order to pebble the outputs.

Based on the decomposition of $FFT[d]$ given in Lemma 1, it can easily be seen that pebbling the 2^j-S trees contained in $\{A_\ell[d-j]\}$ implies that 2^j-S outputs of each subgraph $B_m[j]$ must be pebbled starting from an initial configuration of at most S pebbles on $\{B_m[j]\}$. The following lemma gives a general lower bound on the number of moves required to do this.

Lemma 4: Let $\{FFT_i[k] \mid 0\leq i<2^{d-k}\}$ be a set of 2^{d-k} distinct FFT graphs, each on 2^k inputs, and assume that at least 2^{k-c+2} outputs in each graph must be pebbled using space $S\leq 2^{d-c}$ (c is a small positive integral constant) and starting from an initial configuration of at most S pebbles on the graphs. Then, $\Omega(k\cdot 2^d)$ moves are required to pebble these outputs.

Proof: Assume that each FFT graph $FFT_i[k]$ has an initial configuration of S_i pebbles, where $\sum S_i\leq S$. Then, there are at least 2^{d-k-1} FFT graphs for which $S_i\leq 2^{k-c+1}$, since otherwise, at least 2^{d-k-1} graphs have $S_i>2^{k-c+1}$, so that $\sum_i S_i>2^{d-c}$, which contradicts $\sum_i S_i\leq S\leq 2^{d-c}$. The number of moves required to pebble such an FFT graph is at least the size of the subgraph that results from deleting vertices initially holding pepbbles and output vertices that do not have to be pebbled. In Carlson [2], it is shown that this subgraph has size $\Omega((k-c+1)2^{k-c+1})$ since it is the result of deleting at most 2^k-2^{k-c+2} output vertices and 2^{k-c+1} vertices holding pebbles. Thus, the total number of vertices in $\{FFT_i[k]\}$ that most be pebbled (and hence, the total number of moves that must be made) satisfies

$$T \geq \sum_{0\leq i<2^{d-k-1}} T_i \qquad \text{where } T_i = \Omega((k-c+1)2^{k-c+1})$$

$$\begin{aligned} \text{Thus,} \quad T &= 2^{d-k-1}\,\Omega((k-c+1)2^{k-c+1}) \\ &= \Omega((k-c+1)2^{d-c}) \\ &= \Omega(k\cdot 2^d) \end{aligned}$$

since c is a small positive constant independent of k and d.Q.E.D.

To complete the proof of Theorem 1, Lemma 4 implies that $\Omega(j\cdot 2^d)$ moves are required to pebble 2^j-S outputs in each subgraph $B_m[j]$, and this work must be done in the course of pebbling the output set Y. Since there are $\Omega(2^d/2^j)$ such output sets for which the same argument holds, we have a lower bound of $T=\Omega(j\cdot 2^{2d}/2^j) = \Omega(n^2(\log_2 S)/S)$ moves required to pebble the outputs of an n-input FFT graph in an order defined by the bit reversal permutation. Q.E.D.

We now present a pebbling strategy that comes close to meeting the lower bound of Theorem 1.

Proposition 1: The outputs of the FFT graph FFT[d] can be pebbled in an order based on the bit reversal permutation using S pebbles and $T=O(n^2(\log_2 S)(\log_2 n-\log_2 S)/S)$ moves.

Proof: Assume that $2^{j+1}\leq S<2^{j+2}$. To pebble a set of outputs $Y = \{y_{B(c)},\ldots,y_{B(c+2^j-1)}\}$, we must pebble the roots of 2^j distinct trees contained in $\{A_\ell[d-j]\}$. Since each tree has a space requirement of d-j+1, at most S/(d-j+1) such trees can have the last open path from the root to an input vertex of FFT[d] closed at the same time by a pebble placement somewhere on $\{B_m[j]\}$. Thus, we pebble subsets of Y, each containing $2^j/(d-j+1)$ outputs that must be pebbled in consecutive order by pebbling the trees rooted at these outputs in a synchronous fashion. It can easily be seen that in order to pebble such a set of trees in a synchronous fashion, it suffices to pebble all outputs of $\{B_m[j]\}$, which can be done in $O(j\cdot 2^d)$ moves using a level-by-level strategy on each subgraph $B_m[j]$ (note that pebbling the outputs of a single subgraph $B_{m_0}[j]$ allows a set of "synchronous" moves to be made on all trees since each tree has exactly one input that is an output of $B_{m_0}[j]$). The set Y contains O(d-j) subsets, each of which requires $O(j\cdot 2^d)$ moves to pebble all outputs, and there are $2^d/2^j$ such output sets. Thus, it follows that $T = O(j\cdot(d-j)\cdot 2^{2d}/2^j) = O(n^2(\log_2 S)(\log_2 n-\log_2 S)/S)$ Q.E.D.

At this point, we conjecture that the pebbling strategy outlined in Proposition 1 is optimal, i.e. that the lower bound of Theorem 1 can be improved to $T=\Omega(n^2(\log_2 S)(\log_2 n-\log_2 S)/S)$. The analysis in Theorem 1 showed that pebbling a set of 2S consecutively ordered outputs requires $\Omega(n\cdot\log_2 S)$ moves, since a large number of vertices at level $\log_2 S$ in the FFT graph must be pebbled. In addition, the pebbling strategy of Proposition 1 illustrates the fact that at most $S/(\log_2 n-\log_2 S)$ trees of height $\log_2 n-\log_2 S$ rooted at outputs can be pebbled in a synchronous manner, and it is straightforward to show that pebbling a set of $S/(\log_2 n-\log_2 S)$ consecutively ordered outputs also requires $\Omega(n\cdot\log_2 S)$ moves. This would seem to imply a lower bound of $T=\Omega(n^2(\log_2 S)(\log_2 n-\log_2 S)/S)$, however the difficulty in proving such a bound lies in the small size of the output set (recall that in Theorem 1 we assumed that up to S pebbles were initially being held on outputs). Fortunately, the results that have been presented for pebbling the outputs of an FFT graph in a certain order can be applied to a back-to-back FFT graph having a certain connection pattern. This is the topic of the next section.

4. Pebbling Back-to-Back FFT Graphs with Certain Connection Patterns

In this section, we consider the effect that the connection pattern can have on the simultaneous time and space requirements of a back-to-back FFT computation. First of all, we show that for a certain connection pattern, the lower bound $T=\Omega(n^3/S^2+(n^2\log_2 n)/S)$ for an n-input back-to-back FFT [2,8] can be achieved by a pebbling strategy similar to that of Savage and Swamy [6] for any value of space between $\Omega(\log_2 n)$ and $O(n)$. This result has applications to resource efficient algorithms for polynomial multiplication, when done via back-to-back FFTs, the desired pattern arises. We then give an upper bound for a different connection pattern that is of the form $T=O(n^3\log_2(\log_2 n-\log_2 S)/S^2 + (n^2\log_2 n)/S)$. This bound is an asymptotic improvement over pebbling strategies mentioned by Tompa [8], and is within a $\log\log_2 n$ factor of the known lower bound $T=\Omega(n^3/S^2 + (n^2\log_2 n)/S)$ for space $S=O(n^\varepsilon)$. We also conjecture that the above upper bound cannot be improved, i.e. there is a connection pattern for which a matching lower bound can be derived. The key idea that seems to imply the existence of this lower bound is that the connection pattern defines a partial order for pebbling the outputs of a single FFT, for which an analysis similar to that of Theorem 1 applies. Note: proofs of the results presented in this section have been omitted due to limitations on space. See [10] for additional details.

The first connection pattern that we consider is given by the bit reversal permutation; such a back-to-back FFT graph on $n=2^d$ inputs is denoted by BBFFT[d,B], where B(x) is the bit reversal of the d-digit binary representation of x. BBFFT[3,B] is illustrated in Figure 3.

The following proposition states the existence of an optimal pebbling strategy for BBFFT[d,B]. The key idea in this pebbling strategy is that for a set of outputs in $FFT_1[d]$ of size proportional to the amount of available space, the outputs of $FFT_0[d]$ need only be pebbled once, and this can be done using the pebbling strategy of Savage and Swamy [6].

Proposition 2: The outputs of the back-to-back FFT graph BBFFT[d,B] can be pebbled using S pebbles and $T=O(n^3/S^2 + (n^2\log_2 n)/S)$ moves.

As evidenced by the following corollary, the pebbling strategy of Proposition 2 can be applied to polynomial multiplication, yielding time and space efficient algorithms.

Corollary 1: The multiplication of two polynomials, each of degree n, can be performed by a straight-line algorithm that uses S temporary registers and $T=O(n^3/S^2 + (n^2\log_2 n)/S)$ operations.

The second connection pattern that we consider for a back-to-back FFT graph on $n=2^d$ inputs, BBFFT[d,I], is given by the identity permutation $I(x)=x$. We are able to derive an upper bound for pebbling BBFFT[d,I] that is an asymptotic improvement over results of Tompa[7,8] and conjecture that a matching lower bound can be proven for this particular connection pattern.

Theorem 2: The outputs of the back-to-back FFT graph BBFFT[d,I] can be pebbled using S pebbles and $T=O(n^3 \log_2(\log_2 n-\log_2 S)/S^2 + (n^2\log_2 n)/S)$ moves.

Proof (sketch): We employ a Savage and Swamy [6] pebbling strategy on $FFT_1[d]$, and pebble sets of $S/(\log_2 n-\log_2 S)$ binary trees in synchronous fashion. A detailed analysis shows that this can be done by pebbling $\log_2 n-\log_2 S$ subFFTs of size $S/(\log_2 n-\log_2 S)$ in $FFT_0[d]$, but to insure that subtrees in $FFT_1[d]$ are pebbled synchronously, each of these subFFTs must be contained in a distinct subFFT of size $n/(\log_2 n-\log_2 S)$. Thus, a set of S outputs that are pebbled in $FFT_0[d]$ is split among $\log_2 n-\log_2 S$ subFFTs, and an analysis similar to that of Theorem 1 shows that $n\cdot\log_2(\log_2 n-\log_2 S)$ moves are required to pebble such an output set, from which the stated bound follows. For further details see [10]. Q.E.D.

At this point, we conjecture that the upper bound of Theorem 2 cannot be improved, i.e. that when BBFFT[d,I] is pebbled using space S, $T=\Omega(n^3\log_2(\log_2 n-\log_2 S)/S^2 + (n^2\log_2 n)/S)$ moves are required. The key point from which such a lower bound would follow concerns the order in which the inputs of a binary tree must be pebbled. In the pebbling strategy of Theorem 2, we pebble binary trees in $FFT_1[d]$ by pebbling subtrees one after the other, which leads to a certain separation of outputs that are pebbled consecutively in $FFT_0[d]$. However, we cannot guarantee that pebbling a binary tree requires the pebbling of a large number of subtrees one after the other, eventhough such a result seems plausible enough. If this were the case, a matching lower bound would follow.

References

1. Carlson, D. A., "Time-Space and Size-Space Tradeoffs for Oblivious Computations", Report CS-63 (Ph.D. Thesis), Department of Computer Science Brown University, Providence, RI, September 1980.

2. Carlson, D. A., "Time-Space Tradeoffs for Back-to-Back FFT Algorithms", IEEE Trans. Computers (to appear).

3. Carlson, D. A. and J. E. Savage, "Extreme Time-Space Tradeoffs for Graphs with Small Space Requirements", Info. Proc. Letters, Vol. 14, No. 5, pp. 223-227 (July, 1982).

4. Paterson, M. S. and C. E. Hewitt, "Comparative Schematology", Proc. Proj. MAC Conf. on Concurrent Systems and Parallel Computation, pp. 119-127, June 1970.

5. Paul, W. J. and Tarjan, R. E., "Time-Space Tradeoffs in a Pebble Game", Acta Informatica, Vol. 10, pp. 111-115, 1978.

6. Savage, J. E. and S. Swamy, "Space-Time Tradeoffs on the FFT Algorithm", IEEE Trans. Info. Theory, Vol. IT-24, pp. 563-568, September 1978.

7. Tompa, M., "Time-Space Tradeoffs for Computing Functions Using Connectivity Properties of Their Circuits", J. Comp. Sys. Sci., Vol. 20, pp. 118-132, April 1980.

8. Tompa, M., "Corrigendum: Time-Space Tradeoffs for Computing Functions Using Connectivity Properties of Their Circuits", J. Comp. Sys. Sci., Vol. 23, pp. 106, 1981.

9. Tompa, M., "Two Familiar Transitive Closure Algorithms which Admit No Polynomial Time, Sublinear Space Implementations", SIAM J. Computing, Vol. 11, pp. 130-137, February 1982.

10. Carlson, D. A., "Time-Space Tradeoffs for Back-to-Back FFTs: the Effect of the Connection Pattern", University of Massachusetts Technical Report, in preparation.

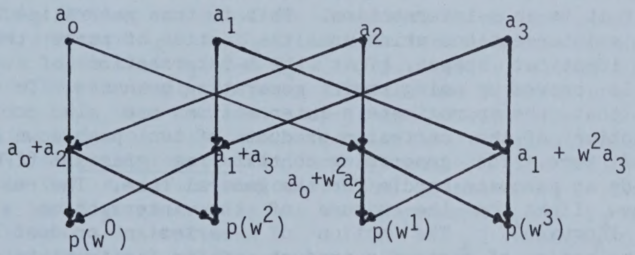


Figure 1: FFT[2].

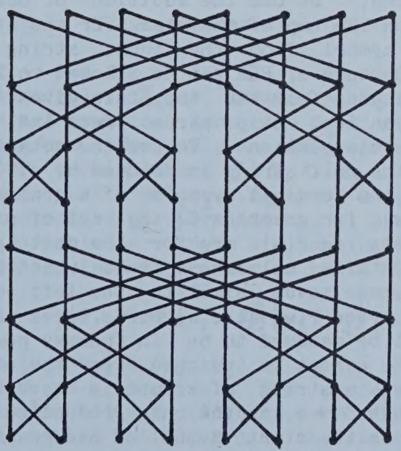


Figure 2: BBFFT[3, I].

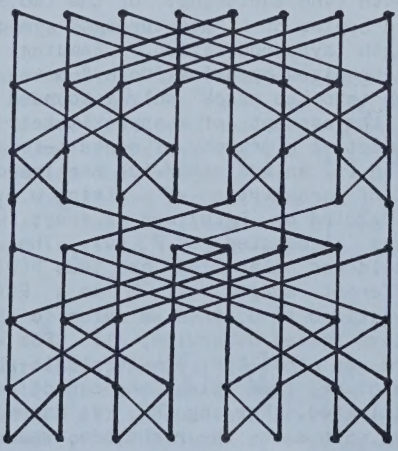


Figure 3: BBFFT[3, B].

CONTEXT-FREE INTERSECTION OF CONTEXT-FREE LANGUAGES

SUKHAMAY KUNDU
Computer and Information Sciences Department
University of Florida, Gainesville, FL 32611, USA

ABSTRACT

It is known that the intersection of two context-free languages (CFL) is in general not context-free. We show that a restricted form of the intersection is always context-free and that every context-free intersection is a restricted type of intersection. The restricted intersection is dependent on the defining grammars of the CFLs. A generalization of this result shows that certain bounded approximation to the restricted intersection is also context-free. These results shed new light on the nature of the intersection problem of CFLs.

1. INTRODUCTION

The main result of this paper gives a characterization of the context-free intersection of two context-free languages (CFL). The concept of s-intersection, introduced here, is based on parse trees of the strings in the language and it is shown that the intersection of CFLs is context-free if and only if it is an s-intersection. This is then generalized to form a hierarchy of s-intersections which uses the notion of parse trees having approximately identical shapes. That the s-intersection of two CFLs is context-free is proved by using their generating grammars. To prove the general case, that the approximate s-intersections are also context-free, we use the notion of the cartesian product of two push-down automatons (PDA), defined here. The generating context-free grammars (CFG) do not seem to provide an adequate handle for the general case. The results given here shed new light on the nature of the intersection problem of context-free languages. The notion of cartesian product of PDAs generalizes the notion of cartesian product of two finite state automaton. The main obstacle in defining the product PDA is the synchronization of growth (and shrinking) of the two stacks. We use the addition of one or more copies of a "see through" symbol on the top of the stack for the stack length synchronization. Reading a symbol from the input string is synchronized by using "delay" moves. Because a PDA can be allowed to look down into the stack only a bounded distance from the top, this gives rise to the concept of k-approximately identical shape parse trees and the respective hierarchy of context-free s-intersections. We use the notations as in [1] unless otherwise mentioned; the null string is denoted by e. The set of parse trees of a string w over the terminal symbols of a grammar G is denoted by $P_G(w)$, or in short, $P(w)$; for grammars G_i the set of parse trees is denoted by $P_i(w)$. The trees in $P(w)$ are for the most part considered unlabelled and thus $P(w)$ contains only those trees which have different shapes. If M is a PDA whose moves simulate the left most deviations in G, then we refer to the trees $P(w)$ also as parse trees in M. Unless stated otherwise, all CFGs will be assumed to be in Chomsky normal form and that the non-terminals which cannot be reached from the start symbol or from which one cannot derive a string of terminals have been eliminated. For any CFG, we assume that $S \rightarrow e$ is the only production (if any) with e on the rightside, where S is the start symbol of the grammar, and in that case S does not occur on rightside of a production. We write $M_1 \supseteq M_2$ to indicate that each transition of the PDA M_2 is also a transition of the PDA M_1.

A parse tree for a string w of terminals has the property that each node has exactly two children unless it is adjacent to a terminal node. The tree obtained by eliminating the terminal nodes of such a tree is called a 2-tree. This reduced tree contains all the information about the original parse tree, except perhaps the terminal string itself. This slight shortcoming is easily removed if we assume that for each non-terminal A there is at most one production of the form $A \rightarrow a$, a is a terminal. The original grammar can be modified to meet this property as follows. Replace the rules $A \rightarrow a_1 \mid a_2 \mid \ldots \mid a_k$ by $A_i \rightarrow a_i$, $1 \leq i \leq k$ by introducing the new non-terminals A_i. A rule $A \rightarrow BC$ is replaced by $A_i \rightarrow BC$, $1 \leq i \leq k$; similarly, the rules containing A on the right as in $B \rightarrow AC$ or $B \rightarrow CA$ are replaced by $B \rightarrow A_iC$ and $B \rightarrow CA_i$, $1 \leq i \leq k$, respectively. The new grammar generates the same strings via the parse trees of the same structures as the original grammar. We often refer to a reduced parse tree of the string w as its parse tree; this terminology is used in Section 3 onwards for the purpose of defining the shape of a tree. The reduced parse tree of a single character "a" is the tree with one node labelled S, the start symbol. In particular, two such parse trees cannot be distinguished. In what follows, it will be convenient to assume that the CFLs contain no strings of length ≤ 1.

2. s-INTERSECTION AND THE MAIN RESULT

Let L_1 and L_2 be two context-free languages which are generated by the context-free grammars G_1 and G_2. Let L be the language which consists of all strings w such that $P_1(w) \cap P_2(w) \neq \phi$, i.e., w belongs to both $L(G_1)$ and $L(G_2)$ and has at least one parse tree in each G_i which are of *identical shape*. This is same as to say that, except for the different non-terminals, w has at least one common left-most derivation in G_1 and G_2. L is called the *s-intersection* (s for syntactic). The definitin of L is dependent on the specific choice of the grammars G_i. Two different grammars G'_1 and G'_2 generating the same languages L_1 and L_2 may give rise to a different s-intersection L'. If $L(G_1) = L(G_2)$, then the s-intersection consists of those strings in the language which allow a common syntactic interpretation under the two grammars.

Theorem 1. The s-intersection of two CFLs is context free.

Proof. Let $G_i = (V_i, \Sigma_i, R_i, S_i)$, where V_i are the variables or non-terminals, Σ_i the terminals, R_i the productions or rules, and S_i are the start symbols. Here, the G_i's are arbitrary and need not be in Chomsky normal form. We define a new grammar G whose variables are $V = V_1 \times V_2$; and the terminals are $\Sigma = \Sigma_1 \cap \Sigma_2$. We define the productions P of G as follows. Let $A \rightarrow u_1u_2 \ldots u_m$ and $B \rightarrow v_1 v_2 \ldots v_t$ be two productions in G_1 and G_2, respectively, where (1) m = t, and (2) for each i either both u_i and v_i are variables or they are terminals and $u_i = v_i$. The corresponding production in G is defined to be

$$(2.1) \qquad (A, B) \rightarrow [u_1, v_2][u_2, v_2] \ldots [u_t, v_t]$$

where $[u_i, v_i] = (u_i, v_i)$ if u_i, v_i are variables, and $[u_i, v_i] = u$ if $u_i = v_i = u$ is a terminal. The production (2.1) is defined only when the conditions (1) and (2) hold. P consists of all productions of the form (2.1); we let $S = (S ,S)$ be the start symbol of G. If both $S_1 \rightarrow e$, $S_2 \rightarrow e$ are in the respective grammars, then $S \rightarrow e$ is in G.

Now, given a string $w \in L(G)$ and a parse tree T of w, the first components of the labels of the nodes of T define a parse tree of w in G_1 and the second components define a parse tree in G_2. The shape of both the parse trees are identical to that of T itself. Conversely, if T_1 and T_2 are two identical shape parse trees of a string w, respectively, in G_1 and G_2, then the tree T obtained by combining the labels at the corresponding nodes of T_1 and T_2 give a parse tree of w in G. Thus $L(G)$ equals the s-intersection of L_1 and L_2, and the theorem is proved.

Theorem 1 can be regarded as a generalization of the result which states that the set intersection of two regular languages is regular. The parse tree for any string in a regular language has only one shape, provided the grammar is in right regular form, say, and is normalized to exclude productions of the type $A \rightarrow B$, where both A, B are non-terminals, and exclude productions $A \rightarrow e$, if A is not the start symbol. Thus the ordinary set intersection $L_1 \cap L_2$ is identical to the s-intersection for regular grammars, and is not dependent otherwise on the particular generating grammars G_i. Theorem 1 provides a different way of looking at the intersection problem of languages.

Example. Let G_1 and G_2 be the CFGs given by

$$G_1 = (\{S_1, A, B\}, \{a, b\}, \{S_1 \rightarrow AB \mid AS_1B, A \rightarrow AA, A \rightarrow a, B \rightarrow b\}, S_1)$$

$$G_2 = (\{S_2, C, D\}, \{a, b\}, \{S_2 \rightarrow CD \mid CS_2D, D \rightarrow DD, C \rightarrow a, D \rightarrow b\}, S_2)$$

Then, $L(G_1) = \{a^m b^n : m \geq n \geq 1\}$ and $L(G\ ) = \{a^m b^n : 1 \leq m \leq n\}$. The set intersection $L(G_1) \cap L(G_2) = \{a^n b^n : n \geq 1\}$ is in this case the same as the s-intersection and is context-free by Theorem 1. The various parse trees for w = aabb in G_i and G are shown in Fig. 1. The next corollary shows that a context-free set intersection of two CFL's is necessarily an s-intersection for suitable choices of the generating CFGs. In particular, the concept of s-intersection is more powerful than set intersection.

Corollary 2. If L_1 and L_2 are two CFLs such that the set intersection $L = L_1 \cap L_2$ is context-free, then there are CFGs G_1 and G_2 such that $L_i = L(G_i)$, $i = 1, 2$ and L equals the s-intersection of L_i's with respect to the grammars G_i's.

Proof. Let $G = (V, \Sigma, R, S)$ and $G'_i = (V_i, \Sigma_i, R_i, S_i)$, $i = 1, 2$, such that $L = L(G) = L_1 \cap L_2$ and $L(G'_i) = L_i$. Define $G_i = (V_i \cup V, \Sigma_i \cup \Sigma, R_i \cup R \cup \{S \rightarrow S_i\}, S)$. Assuming that V is disjoint from each V_i, we have $L(G_i) = L(G) \cup L(G'_i) = L_i$. Each string w in L now allows a parse tree, namely, one in G which occurs among the parse trees under G_1 and G_2. These are also the only strings which have identical shape parse tree(s) in G_i's.

The following two theorems are stated without proof. The proofs are similar to that of Theorem 1, except that for Theorem 4 one has to use the concept of acyclic digraphs to capture the notion of syntax structure of a context sensitive language (CSL). In the remaining sections, we generalize Theorem 1 to the case where the parse trees have approximately identical shapes.

Theorem 3. If G is regular, then the s-intersection of a CFL and $L(G)$ is also regular.

Theorem 4. The s-intersection of two CSL is context-sensitive.

3. SHAPE VECTOR OF A PARSE TREE

A 2-tree T is a binary tree where each non-terminal node has exactly two children. A 2-tree with n nodes has $t = (n+1)/2$ terminal nodes, n being odd. We define the shape number of a node to be the number of left branches on the (unique) path from root to that node. The shape number does not account for the right branches present on the path or how they are interspersed with the left branches. The shape vector of a 2-tree is defined to be the vector $s = (s_1, s_2, \ldots, s_t)$, where s_i is the shape number of the ith terminal node in the order left-to-right. See Fig. 2. The shape vector of the tree with only one node is (0). The proof of Lemma 5 is by induction on the number of nodes and is omitted here.

Lemma 5. The shape vector of a 2-tree determines the tree uniquely.

Two 2-trees are said to have approximately identical shape upto a number k if they have the same number of nodes and the shape numbers s_i and s'_i of corresponding terminal nodes in the left to right order differ by at most k, i.e.,

$$|s_i - s'_i| \leq k \text{ for } i = 1, 2, \ldots, t.$$

We say, in short, that the trees are k-ids. Two 2-trees have the identical shape if they are 0-ids, and any two 2-trees with t terminal nodes are (t-1)-ids.

Given a 2-tree which is a parse tree of an input string w, an alternate view of the shape numbers is obtained by considering the processing of w by the PDA associated with the underlying Chomsky normal form grammar G. Let M(G) be the PDA that simulates the leftmost deviations in G, defined [1, page 113] by the transitions (3.1) - (3.3) shown below. M(G) has only 2 states, s the starting state and f the final state; the stack symbols are the non-terminals of G. S is the start symbol of the grammar in (3.1).

(3.1) ((s, e, e), (f, S)), the initial move
(3.2) ((f, e, A), (f, BC)), for each rule $A \rightarrow BC$ in the grammar, and
(3.3) ((f, a, A), (f, e)), for each rule $A \rightarrow a$.

Fig. 3(i) shows in solid line the reduced form of a parse tree of the string "ahid". Fig. 3(ii) shows the stacks at various stages of processing by the PDA using moves corresponding to the parse tree. The shape number of the node h (H) equals 2, which is one less than the stack size when the symbol h is ready by the PDA. The following theorem is proved easily.

Theorem 6. The shape number of a node in a 2-tree is one less than the size of the stack when the associated node (or, more precisely, its label) is on the top of the stack, if it is an internal node, or when the input symbol associated with that node is read from the input, if it is a terminal node.

4. CARTESIAN PRODUCT OF PDAs

Let $M = (K_i, \Sigma_i, \Gamma_i, \Delta_i, s_i, F_i)$, $i = 1, 2$, be two PDAs where K_i is the set of states, Σ_i the input symbols, Γ_i the stack symbols, Δ_i the transitions, $s_i \in K_i$ the starting state, and $F_i \subseteq K_i$ is the set of final states (all sets are finite). Without loss of generality, we assume that each transition reads at most one input symbol. If $u = u_1 u_2 \ldots u_i$ and $v = v_1 v_2 \ldots v_i$ are two strings of the same length, let u x v denote the string of symbol pairs $(u_1,v_1)(u_2,v_2)\dots(u_i,v_i)$. We define the cartesian product PDA $M = M_1 \times M_2 = (K, \Sigma, \Gamma, \Delta, s, F)$ as follows:

$$K = K_1 \times K_2, \quad \Sigma = \Sigma_1 \cap \Sigma_2, \quad \Gamma = \Gamma_1 \times \Gamma_2$$

$$\begin{aligned}\Delta = \{(((p_1, p_2), a, \alpha), ((q_1, q_2), \beta)):\ & ((p_1, a, \alpha_1), (q_1, \beta_1)) \in \Delta_1, \text{ and } ((p_2, a, \alpha_2), (q_2, \beta_2)) \in \Delta_2, \\ & \alpha = \alpha_1 \times \alpha_2 \text{ and } \beta = \beta_1 \times \beta_2, a \in \Sigma \cup \{e\}\}\end{aligned}$$

$$s = (s_1, s_2), \text{ and } F = F_1 \times F_2$$

The language defined by a PDA is the set of all strings w which take the machine from the starting state to one of the final states and resulting in empty stack. The processing of the input begins with the empty stack. The language defined or accepted by the machine M_i can be then written as

$$L(M_i) = \{w \in \Sigma_i^*: (s_i, w, e) \vdash^* (p, e, e) \text{ for some } p \in F_i\}$$

Let σ be a sequence of moves in the product machine M. Then σ defines two parallel sequences of moves, σ_1 and σ_2, one for each of the machines M_1 and M_2. The state of M_i after the jth move is simply the ith coordinate of the state of M at that point. Similarly, the stack of M_i is given by the ith coordinates of the stack symbols of M, taken in the same order. We say σ simulates in parallel the sequences σ_1 and σ_2. These two sequences of moves in M_1 and M_2 have the properties that (1) they read an input symbol at the same time, (2) an equal number of stack symbols are removed and an equal number of symbols are put back on stack in each move, and following each move (3) the stacks in the two machines contain the same number of symbols. The property (3) is actually a consequence of (2). On the other hand, if we take any two sequences of moves, one for each machine M , processing the same input string and which have the properties (1) - (3), then they can be simulated by a sequence of moves in the product M.

In order to facilitate the description of the language accepted by the product PDA M in terms of those of M_1 and M_2, we assume without loss of generality that each M_i and thus M looks at most one stack symbol at a time. Consider the context-free grammars G_i that is associated with the PDAs M_i [1, page 117]. The productions in G_i simulate the transitions in the machine M_i and generate the same language as accepted by M_i. If the sequences σ_1 and σ_2 of moves which satisfy the properties (1) - (3) and correspond to leftmost derivations according to parse trees T_1 and T_2, respectively, in G_1 and G_2, then T_i's have identical shapes. Conversely, let T_i be a parse tree in G_i, i = 1, 2 which are of identical shape. Then the move sequences σ_i in the machines M_i which corresponds to the leftmost deviations obtained from T_i have the properties (1) - (3). We summarize these observations in the following theorem.

Theorem 7. Let M_1 and M_2 be two PDAs and M be their cartesian product such each PDA looks at at most one stack symbol at a time. Then $L(M) = \{w \in L(M_1) \cap L(M_2): P_1(w) \cap P_2(w) \neq \emptyset\}$.

5. SYNCHRONIZATION OF PDAs

Let M be a PDA and k, a fixed positive integer. Let M(k) be the PDA defined by the transitions (5.1) - (5.3); M and M(k) have the same states, the same starting state, the input alphabet, etc., except that M(k) contains a new stack sumbol $\emptyset$.

(5.1) $((p, e, e), (P, \emptyset)), ((p, e, \emptyset), (p, e))$ for each state other than the starting state

(5.2) $((p, e, e), (p, e))$ for each state other than the starting state

(5.3) $((p, a, \emptyset^{i}\alpha), (q, \emptyset^{j}\beta))$ for each transition $((p, a, \alpha), (q, \beta))$ in M, $a \in \Sigma \cup \{e\}$, where $|\alpha| + i \leq k$ and $j \leq i$.

It is easy to see that $M(k) \supseteq M(k')$ if $k > k'$. The following lemma is stated without proof for want of space as are Lemma 9 and Theorem 10.

Lemma 8. If M is the PDA defined by (3.1) - (3.3) corresponding to a Chomsky normal form CFG G and $k \geq 1$, then the PDA M(k) defined by (5.1) - (5.3) from M accepts the same language as that of M. Moreover, a sequence of moves in M(k) which accepts an input string w defines a unique parse tree of w in the grammar G.

The transitions (5.2) helps introduce arbitrary amount of time delay (without affecting the stack) for the purpose of synchronizing the reading of an input symbol by two PDAs at the same instant. These are called delay transitions. The transitions (5.3) and (5.1) together help maintain the stack lengths without affecting a potential move. The flexibility is however limited by the constant k. A move that uses transition of type (5.1) is called a dummy move.

We illustrate the use of a construction like M(k) in synchronizing two PDAs. Figs. 4(i) - (ii) show two parse trees of a string w = "ghdkmjf". Let M_1 and M_2 be the associated PDAs defined by (3.1) - (3.3) from respective underlying CFGs and subsequently modified by (5.1) - (5.3) with k = 3. Fig. 4(iii) shows a parallel sequence of moves in M_1 and M_2 each accepting the string w. In step 4, M_1 adds the sumbol G to its stack while M_2 adds $\emptyset$ to its stack bringing both stacks to the same length (=4). M_2 also introduces in this step the needed delay to synchronize reading of the first input symbol g in step 5 with that of M_1. In step 5, M_1 removes G from its stack and M_2 replaces $\emptyset G'$ by $\emptyset$. In step 6, M_1 uses a delay transition and in step 7 it uses a move similar to that of M_2 in step 4. During this time M_2 expands its stack in keeping with the length of that of M_1, preparing to read h, which takes place in step 8 for both M_1 and M_2. In steps 11 and 12, M_2 introduces two $\emptyset$'s to its stack including two units of delay before reading the input symbol k in step 13 together with M_1. Both PDAs accept the input in step 19. One important observation to make here is that the two $\emptyset$'s on the stack of M_2 in step 13 corresponds to the fact that the shape number of the node K', associated with the input symbol k read at that step, in the second parse tree is two less than that of the node K in the first parse tree. A similar phenomenon occurs in each of the steps 5 and 15, this time the difference being one. Next, the extra $\emptyset$ in the stack of M_1 in step 8 is due to the fact that the node H (for the input symbol h) has one less shape number in the first parse tree than in the second. A similar phenomenon occurs again in step 9.

Lemma 9. Let G_i, i = 1, 2 be two CFGs in Chomsky normal form, $M_i = M(G_i)$, and $T_i \in P_i(w)$ for some input string $w = w_1 w_2 \dots w_n$. If the shape vectors of T_1 and T_2 are k-ids, then w is accepted by $M_1(k+1) \times M_2(k+1)$.

6. THE APPROXIMATE s-INTERSECTION

We now state the result on the context-free property of bounded approximate s-intersections of two context-free languages.

Theorem 10. Let G_i, i = 1, 2 be CFGs in Chomsky normal form and $k \geq 0$. Then the strings which allow at least one parse tree in each G_i such that their shape vectors differ coordinate-wise by at most k form a context-free language.

Corollary 11. Let G_i, i = 1, 2 be as in Theorem 13. Let $L_k = \{w \in L(G_1) \cap L(G_2)$: w has parse trees in G_i which are k-ids$\}$. Then $L_0 \subseteq L_1 \subseteq L_2 \subseteq \cdots$ and $\bigcup_k L_k = L(G_1) \cap L(G_2)$.

Proof. Given any string $w \in L(G_1) \cap L(G_2)$ and two parse trees T_1 and T_2, one in each G_i, the trees are k-ids for a sufficiently large k ($k = |w| - 1$ suffices). Thus $w \in L_k$.

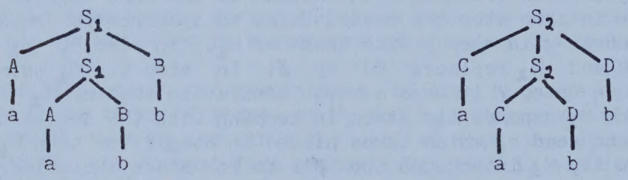
REFERENCES

1. Papadimitriou, C. H. and Lewis, H. R., *Elements of the Theory of Computation*, 1981, Prentice-Hall, Inc., Englewood Cliffs, NJ.

2. Harrison, M. A., *Introduction to Formal Language Theory*, 1978, Addison-Wesley, Reading, MA.

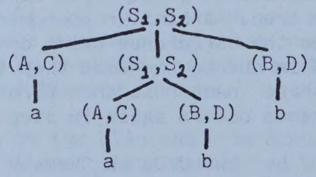
3. Aho, A. V. and Ullman, J. D., *The Theory of Parsing, Translation, and Compiling*, Vol. I, 1972, Prentice-Hall, Inc., Englewood Cliffs, NJ.

4. Ginsberg, S., *The Mathematical Theory of Context-free Languages*, 1966, McGraw-Hill, NY.



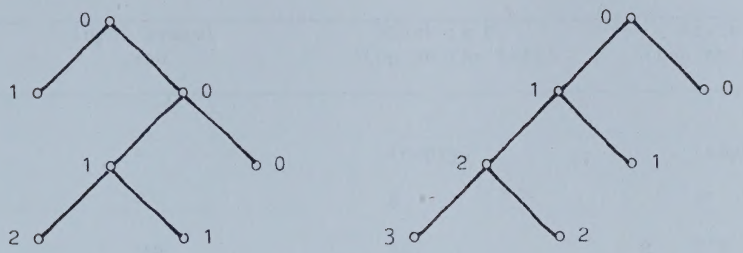
(i) Parse tree in G

(ii) Parse tree in G



(iii) Parse tree in G

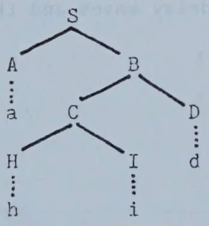
Fig. 1. The parse trees of w = "aabb" in grammars G , G , and G.



Shape vector
= (1, 2, 1, 0)

Shape vector
= (3, 2, 1, 0)

Fig. 2. Two 2-trees and their shape vectors; the shape number of each node appears next to it.

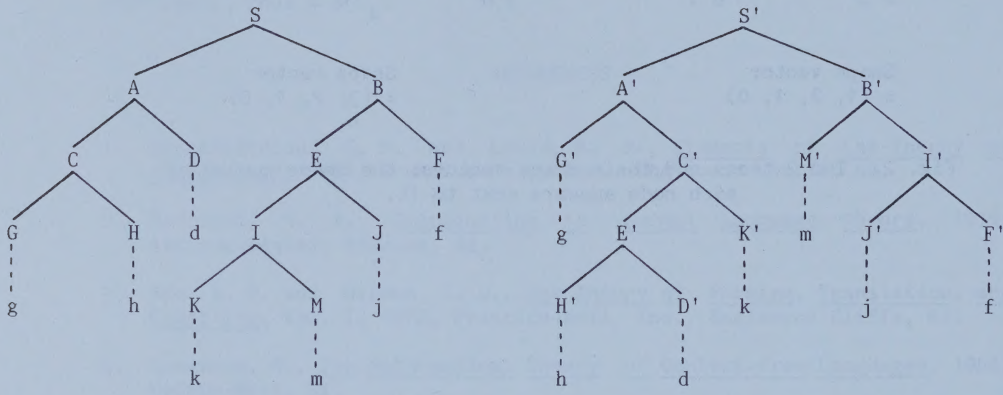


(i) A parse tree of "ahid"

Shape number	Input symbol read	Stack (top on the left)
-	-	(empty)
-	-	S
1	a	AB
-	-	B
-	-	CD
2	h	HID
1	i	ID
0	d	D
-	(accepted)	(empty)

(ii) Stack configurations in processing of "ahid"

Fig. 3. Correspondence between shape numbers and stack size in PDA.



S
A
B
C
D
E
F
G
H
d
I
J
f
g
h
K
M
j
k
m
S'
A'
B'
G'
C'
M'
I'
g
E'
K'
m
J'
F'
H'
D'
k
j
f
h
d

(i) A parse tree for "ghdkmjf" in a grammar G_1

(ii) A parse tree for "ghdkmjf" in another grammar G_2

Fig. 4. Illustration of PDA synchronization by use of delay moves and the see-through symbol ϕ.

Step #	Input symbol read	Stack in M_1 (top on the left)	Stack in M_2 (top on the left)
1.	-	(empty)	(empty)
2.	-	S	S'
3.	-	AB	A'B'
4.	-	CDB	G'C'B'
5.	g	GHDB	ϕG'C'B'
6.	-	HDB	ϕC'B'
7.	-	HDB	E'K'B'
8.	h	ϕHDB	H'D'K'B'
9.	d	ϕDB	D'K'B'
10.	-	ϕB	K'B'
11.	-	EF	K'B'
12.	-	IJF	ϕK'B'
13.	k	KMJF	ϕϕK'B'
14.	-	MJF	ϕϕB'
15.	m	MJF	ϕM'I'
16.	-	JF	ϕI'
17.	j	JF	J'F'
18.	f	F	F'
19.	(accepted)	(empty)	(empty)

(iii) Synchronized processing of "ghdkmjf"; the stacks shown are those prior to the moves.

Fig. 4. (Con't)

FINITE AUTOMATA WITH SEVERAL INITIAL STATES

P. A. S. VELOSO
Deptº Informática, PUC-RJ
22453 Rio de Janeiro, RJ ; Brazil

ABSTRACT

A regular language can have several non-isomorphic minimal finite automata with more than one initial state. The property of having an infinite collection of non-isomorphic minimal finite automata with several initial states is shown to characterize the infinite regular languages.

SUMMARY

The uniqueness (up to isomorphism) of minimal machines depends heavily on their having only one initial state. In fact, Valk |5|, presented an infinite family of pairwise non-isomorphic minimal machines computing the same relation. Here we consider the analogous question for finite automata and their languages. We show that infinite regular languages have infinitely many minimal (i.e. reduced and connected) finite automata with several initial states, whereas a finite language has one such recognizer that contains all the others, up to isomorphism.

A non-initialized finite automaton (nifa, for short) $M = <\Sigma,S,F,I,F>$ is just like the familiar finite automaton |4|, except that it has a nonempty set I of initial states instead of a single initial state. The language accepted by a state $s\in S$ is the set $A(s) = \{w\in\Sigma^* | f(s,w)\in F\}$. We say that M is a nifa for a language L Σ^* iff $L = \bigcup_{s\in I} A(s)$. (Alternatively, one can regard a nifa as a nondeterministic finite automaton where all the transitions are deterministic, the only source of non-determinism lying in the choice of the initial state). Notice that a (deterministic) finite automaton (abbreviated fa) is a nifa with $|I|=1$. Another special case is the multiple-entry finite automaton |2|, which is a nifa with I=S.

In general, we shall adopt for nifa's the usual terminology employed for fa's, with the obvious modifications. For instance, a connected nifa is one with $f(I\times\Sigma^*)=S$; a homomorphism of nifa's has to map all the initial states of the one into the set of initial states of the other. Similarly for reduced nifa, equivalence, subnifa, etc.; cf. |2| or |3|.

Let L denote a given regular language over the alphabet Σ. We shall denote by D(L) the set of all the derivatives $D_u K = \{v\in\Sigma^*/uv\in K\}$ |1| of subsets $K\subset L$ with respect to words $u\in\Sigma^*$ and by d(L) the cardinality of

D(L).

Lemma. Let M be a minimal nifa for L.

(a) If L is finite then M has at most d(L) states.

(b) If L is infinite then M is a proper subnifa of another minimal nifa for L, which is not equivalent to M.

Now, we have some characterizations for the finite languages in terms of their minimal nifa's.

Theorem. The following properties are equivalent.

(1) L is finite.

(2) L has only finitely many minimal pairwise non-equivalent nifa's.

(3) L has a minimal nifa into which every minimal nifa for L is embeddable.

(4) L has only finitely many minimal pairwise non-isomorphic nifa's.

The above result shows that a finite language has a very special minimal nifa. Let us call a maxmin nifa for L a minimal nifa for L into which any other minimal nifa for L can be embedded. A well-known method |1| uses derivatives to construct the minimal fa for a regular language. The next result adapts this method to provide an explicit description of the maxmin nifa's for finite languages.

Corollary. If L is finite then the nifa $N = \langle\Sigma,T,g,I,G\rangle$ with $T = D(L)$, $g(J,a) = D_a J$ for $J \varepsilon T$ and $a \varepsilon \Sigma$, $I = \{J\varepsilon T / J \subset L\}$ and $G = \{J \varepsilon T / \lambda \varepsilon J\}$ is the (unique, up to isomorphism) maxmin nifa for L.

ACKNOWLEDGEMENTS

Helpful discussions with A.Gill as well as financial support from CNPq and COPPE-UFRJ are gratefully acknowledged.

REFERENCES

1. Brzozowski, J.A., Derivatives of regular expressions, J. ACM 11, 481-494 (1964).
2. Gill, A. and Kou, L. T., Multiple-entry finite automata, J. Comput. System Sci. 9, 1-19 (1974).
3. Ginzburg, A., Algebraic Theory of Automata, Academic Press, New York, 1968.
4. Rabin, M.O and Scott, D., Finite automata and their decision problems, IBM J. Res. Develop. 3, 114-125 (1959).
5. Valk, R., Minimal machines with several initial states are not unique, Inform. Contr. 31, 193-196 (1976).

INCREMENTAL UNIFICATION IN EQUATIONAL THEORIES

J.P. JOUANNAUD, C. KIRCHNER, H. KIRCHNER
Centre de Recherche en Informatique de Nancy
Campus Scientifique - B.P. 239 - 54506 Vandoeuvre Les Nancy Cedex
France

ABSTRACT

The subject of this paper is the unification problem (or equations solving) in equational theories in which the set of axioms is split in a set of equations and a set of rewrite rules. We use a generalization of the narrowing process in this frame and propose sufficient conditions to obtain complete sets of unifiers.

1. INTRODUCTION

In this paper, we address unification problems that arise in equational theories. This is known to be an important, however rather hard problem. Applications may be found in automatic theorem proving especially, but also in proving strong consistency of abstract data types by the use of extensions of the Knuth and Bendix completion algorithm [H&H,80].

Many important equational theories have been studied in the past, and specific unification algorithms are actually known for some of them, including commutativity, associativity [PLO,72], idempotence [R&S,78], associativity and commutativity [STI,81], associativity, commutativity and idempotence [L&S,77] and abelian group theory [LAN,79]. All of these algorithms are complete. Some of them are not proved to be finitely terminating.

In a recent work [FAY,79], Fay described a complete unification algorithm for some equational theories that possess a canonical and finite term rewriting system as defined by Knuth and Bendix [K&B,70]. This method relies on using the narrowing process defined by Lankford [LAN,75]. More recently, Hullot [HUL,80] described an improved version of Fay's algorithm, which both allows to avoid any useless computations and gives sufficient conditions for termination.

Problems may arise in practice, that forbid the use of this elegant technique :

- various equational theories contain axioms that cannot be oriented without loosing the finite termination property (commutative axioms for example).

- the completion of many equational theories generates an infinite set of rules (for example the signed trees theory [KKJ,81; K&K,82]).

- some canonical and finite term rewriting systems do not satisfy Hullot's termination condition : group theory for example with the rule $-(x+y) \rightarrow (-y)+(-x)$.

Against such a situation, the set A of axioms can be split in two sets E of equations and R of rules. This technique was still investigated in [HUL,80] when R is confluent modulo E as defined by Huet [HUE,77&80], with the restriction of left linear term rewriting systems. When R is E-confluent as defined by Peterson and Stickel [P&S,81], Hullot states unproved results which come out to be false. As the previous restriction does not appear in this framework, we intend to obtain correct results by an appropriate generalization of Hullot's tools.

In section 2, we recall basic notions about terms and term rewriting systems. Peterson-Stickel's R,E-relation and E-confluence result are given in section 3 together with a new condition for E-confluence called E-commutation, proved to be decidable using critical pairs. Assuming E-commutation, we describe in section 4 the precise link between R,E-reduction and R,E-narrowing. A complete unification algorithm is then given for the theory RUE, assuming a complete terminating unification algorithm for the theory E. Following Hullot, we define in section 5 an improved algorithm and give sufficient conditions for its termination. In conclusion, we discuss briefly problems encountered with the signed trees theory and show the interest of our results in order to build new extensions of the Knuth and Bendix completion algorithm.

2. PRELIMINARIES

Our definitions and notations are consistant with those of Huet and Oppen [H&O,80], Huet [HUE,77&80] and Hullot [HUL,80].

Definitions 1 : Given a set X of variables and a graded set F of function symbols M(F,X) denotes the free algebra over X. Elements of M(F,X) are called terms. Terms may be viewed as labelled trees in the following way : a term t is a partial application of N^* into FUX such that its domain Dom(t) satisfies :

(1) $e \in \mathrm{Dom}(t)$ where e is the empty word.

(2) $u \in \mathrm{Dom}(t_i) \Rightarrow iu \in \mathrm{Dom}(f(\dots t_i \dots))$ for all i in $[1,n = \mathrm{arity}(f)]$.

Dom(t) is called set of occurrences of t, O(t) denotes the subset of non variable occurrences of Dom(t), V(t) the set of variables of t, t/u the subterm of t at occurrence u, $t[u \leftarrow t']$ the term obtained by replacing t/u by t' in t. □

Definitions 2 : Substitutions s are defined to be endomorphisms of M(F,X) with a finite domain D(s). A substitution s is denoted by $\{(x_1 \leftarrow t_1),\ldots,(x_n \leftarrow t_n)\}$. We define the set I(s) of variables introduced by s as the union of the sets V(s(x)) for all x in D(s). We denote by $s_{|W}$ the restriction of the substitution s to the subset W of X. We denote by $\leq$ the subsumption preorder on M(F,X) defined by $t \leq t'$ iff $t' = s(t)$ for a substitution s called a match from t to t'.
Composition of substitutions s and r is denoted by s.r .
Given a subset V of X, we define $s \leq s'[V]$ iff $s(x) \leq s'(x)$ for all variables x in V which is equivalent to $s' = s''.s[V]$ for some substitution s".
V = X is omitted. □

Definitions 3 : We call axiom or equation any pair (t,t') of terms and write it t = t'. The A-equality $=_A$ is the smallest congruence closed under instanciation and generated by a finite set of axioms A. $|-|_A$ denotes one step of A-equality. □

We are interested in solving equations in equational theories, that is in finding substitutions s such that $s(t) =_A s(t')$. To define a basis of the set of solutions, we extend the preorder on substitutions to equational theories.

Definition 4 : We define $t \leq_A t'$ iff $t' =_A s(t)$ for some substitution s called a A-match from t to t'. Given a subset V of X, we define $s \leq_A s'[V]$ iff $s(x) \leq_A s'(x)$ for all x in V, equivalent to $s' =_A s''.s[V]$ for some s". □

Definitions 5 : Given an equational theory $=_A$, two terms t and t' are said to be A-unifiable iff there exists a substitution s such that $s(t) =_A s(t')$. S is a complete set of A-unifiers of t and t' away from $W \supseteq V = V(t) \cup V(t')$ iff :

(1) for all s in S, $D(s) \subseteq V$ and $I(s) \cap W = \emptyset$ (no conflict between variables)

(2) for all s in S, $s(t) =_A s(t')$

(3) for all unifiers s', there exists a unifier s in S such that $s \leq_A s'[V]$

In addition S is said to be minimal if it satisfies the further condition :

(4) for all s and s' in S, $s \leq_A s'$ implies s = s'.

We denote by SU(t,t',A) the set of all A-unifiers of t and t', by CSU(t,t',A) a complete set of A-unifiers of t and t' and by CMSU(t,t',A) a complete and minimal set of A-unifiers of t and t'. A may be omitted when known or empty.

The A-unification algorithm is complete if it generates a complete set of A-unifiers. We do not require any termination property here. □

Many theorical problems arise in equational theories (word problem,...) that can be approached by the use of rewrite rules that is one-way-equations. Working with rules requires good properties as shown by Knuth and Bendix.

Definitions 6 : a term rewriting system R is a set of pairs $\{g_i \rightarrow d_i \mid 1 \leq i \leq n\}$ such that $V(d_i) \subseteq V(g_i)$. We say that a term t R-reduces at occurrence u to a term t' using the rule $g_i \rightarrow d_i$ and we write $t \dashrightarrow^{R}_{[u,i]} t'$ iff there exists a match s from g_i to t/u and $t' = t[u \leftarrow s(d_i)]$. We may omit R or [u,i].

We denote by **> the derivation relation, that is the reflexive transitive closure of --> and by $=_R$ the generated equational theory.

An irreducible term is said in R-normal form. $\overline{t}$ is the R-normal form of t. □

Definition 7 : A term rewriting system R is said to be canonical iff :

(1) $\dashrightarrow^{R}$ is noetherian (has the finite termination property)

(2) $\dashrightarrow^{R}$ is confluent, that is for all t, t_1, t_2 such that t **> t_1 and t **> t_2 there exists a term t' such that t_1 **> t' and t_2 **> t'. □

Canonical term rewriting systems provide a decision procedure for equational theories because $t =_R t'$ iff $\overline{t} = \overline{t'}$. They also provide a complete unification algorithm for the theory $=_R$, based on the narrowing process :

Definition 8 : We say that t is "R-narrowable" at occurrence u belonging to O(t) and we write $t \overset{\wedge}{\dashrightarrow}{}^{R}_{[u,i,s]} t'$ iff s is a most general unifier of t/u and g_i such that $t' = s(t[u \leftarrow d_i])$. The relation -^-> is called narrowing. □

Theorem 1 [HUL,80] : Given a canonical term rewriting system R, if the equational term t = t', where = is considered here as a binary function symbol, satisfies :

$$t = t' \overset{\wedge}{\dashrightarrow}{}^{R}_{[u_0,i_0,r_0]} t_1 \overset{\wedge}{\dashrightarrow}{}^{R} \dots \overset{\wedge}{\dashrightarrow}{}^{R}_{[u_{n-1},i_{n-1},r_{n-1}]} t_n = t'_n$$

with s as most general unifier of t_n and t'_n, then $s.r_{n-1} \dots r_0$ is a R-unifier of t and t', and a complete set of R-unifiers can be obtained in this way.

A complete R-unification algorithm may thus be designed from this result : it is based on unifying equational terms in the narrowing derivation tree.

3. R/E-REDUCTION R,E-REDUCTION AND E-COMMUTATION

We assume now that the set A of axioms is split into a set E of equations and a set R of rules. To work with, a confluence property is also needed.

Definition 9 : Let $\longrightarrow^{R/E}$ be the relation $=_E . \longrightarrow^{R}$; R is said to be E-canonical iff (1) $\longrightarrow^{R/E}$ is finitely terminating (or R is E-noetherian)

(2) for all t, t_1, t_2 such that $t \;{*}{*}{>}^{R/E}\; t_1$ and $t \;{*}{*}{>}^{R/E}\; t_2$ there exist t_1' and t_2' such that $t_1 \;{*}{*}{>}^{R/E}\; t_1'$ and $t_2 \;{*}{*}{>}^{R/E}\; t_2'$ and $t_1' =_E t_2'$ (E-confluence). □

To decide R/E-reducibility, we use the Peterson and Stickel R,E-relation:

Definition 10 : We say that t is R,E-reducible at occurrence u with the rule $g_i \to d_i$ and we write $t \xrightarrow[{[u,i]}]{R,E} t'$ iff there exists a E-match s from g_i to t/u such that $t' = t[u \leftarrow s(d_i)]$. □

Notice that the R,E-reducibility is decidable if the E-matching is decidable.

Definition 11 : A term t is in R,E-normal form iff t is R,E-irreducible. From now up, $\bar{t}$ denotes the R,E-normal form of the term t.
A substitution s is R,E-normalized iff s(x) is in R,E-normal form, $\forall x$ in D(s). □

In order to link relations R/E and R,E together, Peterson and Stickel define the E-compatibility. We introduce here a stronger relation required later on.

Definitions 12 : R is E-commuting (resp. locally E-commuting) iff $\forall t, t_1$ and t_2 such that $t_1 =_E t \longrightarrow^{R,E} t_2$ (resp. $t_1 \mid\!-\!\mid_E t$), $\exists t'$ such that $t_1 \longrightarrow^{R,E} t' =_E t_2$. □

Schematically :

$$\begin{tikzcd}
t \arrow[r, "{R,E}"] \arrow[d, phantom, "=_E"] & t_2 \arrow[d, phantom, "=_E"] \\
t_1 \arrow[r, "{R,E}"] & t'
\end{tikzcd}$$

Notice that now a term t is R/E-reducible iff it is R,E-reducible.

Proposition 1 : R is E-commuting iff it is locally E-commuting. □

Proof : easy induction on the number of steps of the equality $=_E$. □

To prove the decidability of this property, we use a critical pair technique.

Definition 13 : Given two pairs of terms (g',d') and (g,d) with $V(g') \cap V(g) = \emptyset$, an occurrence u of O(g) and a complete E-unification algorithm, the following set $\{<p,q> \mid p = s(g[u \leftarrow d']),\ q = s(d)$ for all s in $CSU(g', g/u, E)\}$ is a complete set of E-critical pairs of (g',d') on (g,d) at occurrence u. A complete set of E-critical pairs of R is obtained for all possible choices of rules $g' \to d'$ and $g \to d$ and of occurrences u of $O(g)$.

A complete set of E-critical pairs of R/E is obtained for all possible choices of rules $g' \to d'$ of R and axioms $g=d$ of E union E^{-1} and occurrences u of $O(g)$. □

<u>Definition 14</u> : A E-critical pair $<p,q>$ of R/E is E-commuting iff $q \xrightarrow{R,E} \cdot =_E p$. □

It is strongly E-commuting if $q = s(d) \xrightarrow[{[u,i]}]{R,E} \cdot =_E p$ with $u \in Dom(d)$. □

<u>Proposition 2</u> : Assume E is a set of axioms $g=d$ such that g and d are linear and $V(g) = V(d)$. Let C be a complete set of E-critical pairs of R/E. Then

- If R is locally E-commuting, then all pairs in C are E-commuting.
- If all pairs in C are strongly E-commuting then R is locally E-commuting. □

<u>Proof</u> : The $\Rightarrow$ way is obvious. The $\Leftarrow$ way follows nearly Huet's proof [HUE,77 &80]. Let us assume that $t=s(g)$, $t_2=s(d)$ and $t_1=t[u \leftarrow r(d')]$, other cases being straight forward :

- If $u \notin O(g)$, the result follows by linearity hypothesis.
- If u belongs to $O(g)$. Then g/u and g' are unified by sUr and there exists s' in $CSU(g/u,g',E)$ and an associated E-critical pair $<p,q>$ of R/E such that $r'.s' = r \cup s[V(g) \cup V(g')]$ for some substitution r'. By hypothesis $s'(d) = q \xrightarrow{R,E} q' = q[v \leftarrow t'']$, with $v \in Dom(d)$.

Thus $s(d)/v = s(d/v) =_E r' . s'(d/v) =_E r'(s'(d)/v) = r'(q/v)$, and $s(d) \xrightarrow{R,E} s(d)[v \leftarrow r'(t'')] =_E r'(q)[v \leftarrow r'(t'')] = r'(q[v \leftarrow t'']) = r'(q')$.
But $t_1 =_E r'(p) =_E r'(q')$ by hypothesis, achieving the proof. □

This result provides a decision procedure for the E-commutation. Moreover the decision criterium using a critical pair technique, it is possible to design a completion procedure for this property, which is crucial for applications.

Let give examples of equational theories satisfying the E-commutation property :

<u>Examples 1</u> :

1/ $E=\{x+y=y+x\}$ $R=\{x+0 \to x\}$ or $R=\{x+0 \to 0\}$ without E-critical pairs.

2/ $E=\{-(x+y)=(-y)+(-x)\}$, $R=\{(x+0 \to x)$ and $(-0 \to 0)\}$. There is a E-critical pair between the equation and the first rule and we must add the new equation $0+(-x) \to -x$ (using a Knuth and Bendix-like completion algorithm).

3/ The binary signed trees theory [KKJ,81] : $E=\{-(x+y)=(-y)+(-x)$ and $--x = x\}$, $R=\{(y+(-x))+x \to y$ and $x+((-x)+y) \to y\}$.

4/ The signed trees theory (see [K&K,82]).

<u>Definition 15</u> : A E-critical pair $<p,q>$ of R is <u>E-confluent</u> iff

$p \overset{*}{\xrightarrow{R,E}} t$ and $q \overset{*}{\xrightarrow{R,E}} t'$ with $t =_E t'$ for some t and t'.

R is said to be E-closed iff all critical pairs in a complete set of E-critical pairs of R are E-confluent. □

Example 2 : For all the theories of example 1, R is E-closed.

We may now write down a particular version of Peterson & Stickel's theorem:

Theorem 2 : Assume R is E-noetherian and E-commuting. Then it is E-confluent iff it is E-closed. □

4. R,E-NARROWING AND RUE-UNIFICATION

R,E-narrowing is simply defined by using E-unifiers instead of unifiers.

Definition 16 : We say that t is R,E-narrowable at occurrence $u \in O(t)$ and we write $t \overset{\wedge}{\dashrightarrow}{}^{R,E}_{[u,i,s]} t'$ iff s belongs to a complete set of E-unifiers of t/u and g_i away from $W \supseteq V(t) \cup V(g_i)$ with $V(t) \cap V(g_i) = \emptyset$, and $t' = s(t[u \leftarrow d_i])$.
A finite sequence of R,E-narrowing starting from a term is called a R,E-narrowing derivation. □

Notice that $t \overset{\wedge}{\dashrightarrow}{}^{R,E}_{[u,i,s]} t'$ implies $s(t) \dashrightarrow{}^{R,E}_{[u,i]} t'$.

We precise now the link between R,E-narrowing and R,E-reduction :

Theorem 3 : Assume R is E-commuting, E-noetherian and E-closed. Let w_0 be a term and s_0 a R,E-normalized substitution with $D(s_0) \subseteq V(w_0)$. Then for every R,E-derivation starting in $w'_0 =_E s_0(w_0)$, there exists a corresponding R,E-narrowing derivation starting in w_0 such that :

$$\begin{tikzcd}
w'_0 \arrow[r, dashed, "{R,E}", "{[u'_0,i'_0]}"'] & w'_1 \arrow[r, dashed, "{R,E}"] & {} \arrow[r, dashed, "{R,E}", "{[u'_{n-1},i'_{n-1}]}"'] & w'_n \\
w_0 \arrow[u, "=_E", "s_0"'] \arrow[r, dashed, "{\wedge\; R,E}", "{[u_0,i_0,r_0]}"'] & w_1 \arrow[u, "=_E", "s_1"'] \arrow[r, dashed, "{\wedge\; R,E}"] & \cdots \arrow[r, dashed, "{\wedge\; R,E}", "{[u_{n-1},i_{n-1},r_{n-1}]}"'] & w_n \arrow[u, "=_E", "s_n"']
\end{tikzcd}$$

with (1) $\forall j,\ 0 \leq j \leq n$, s_j is R,E-normalized and $D(s_j) \subseteq V(w_j)$.

(2) $s_0 =_E s_j . r'_j [V(w_0)]$ where $r'_0 = Id$, $r'_{j+1} = r_j . r'_j$.

Conversely, for every R,E-narrowing derivation starting in w_0 and for every substitution s_0 such that $r'_n \leq_E s_0 [V(w_0)]$, there exists a R,E-derivation starting in some $w'_0 =_E s_0(w_0)$ and also one starting exactly in $s_0(w_0)$. □

Proof : By induction on n. As a detailed proof is rather long and technical (see [K&K,82]), we only sketch it.
Starting from the reduction, the construction goes from left to right and the induction step is expressed on the following diagram :

We first apply the commutation hypothesis in order to find a reduction on w''_j and then prove that w_j is R,E-narrowable at the same occurrence u_j

$$\begin{tikzcd}
w'_j \arrow[r, "{R,E}", "{[u'_j,i'_j]}"'] \arrow[d, phantom, "=_E"] & w'_{j+1} \arrow[d, phantom, "=_E"] \\
w''_j \arrow[r, "{R,E}", "{[u_j,i_j]}"'] & w''_{j+1} \\
w_j \arrow[u, "s_j"'] \arrow[r, "{\wedge\ R,E}", "{[u_j,i_j,r_j]}"'] & w_{j+1} \arrow[u, "s_{j+1}"', "=_E"]
\end{tikzcd}$$

because s_j is R,E-normalized. If s' satisfies $s'(d_{i_j}) =_E (w''_j/u_j)$ and choosing $D(s') \cap D(s_j) = \emptyset$, $s' \cup s_j$ is an E-unifier of d_{i_j} and (w_j/u_j). So $\exists r_j \in CSU((w_j/u_j), d_{i_j}, E)$ $\exists r$ $r.r_j = s_j \cup s'[V(w_j) \cup V(d_{i_j})]$. s_{j+1} is defined as $r_{|V(w_j)}$ and thus $s_{j+1}(w_{j+1}) =_E w''_{j+1}$.

Starting from the narrowing reduction, the construction goes from right to left : s_n is defined by $s_n.r'_n =_E s_0$, $s_j = s_{j+1}.r_j$ and thus we may construct w''_j and prove that it reduces to w''_{j+1}. The upper part of the diagram is obtained by E-commutation hypothesis. We have to use once more the E-commutation hypothesis to construct now a derivation starting in $s_0(w_0)$ exactly. □

We are now ready to prove the correctness and completeness of the R,E-narrowing process. Let us consider = as a binary symbol and t=t' as a term.

Proposition 3 : Let $t_0=t'_0 \overset{\wedge}{\longrightarrow}{}^{R,E} \dots \overset{\wedge}{\longrightarrow}{}^{R,E} t_n=t'_n$ and r in a $CSU(t_n,t'_n,E)$. Then $r.r'_n$ is a A-unifier of t_0 and t'_0 with r'_n as in theorem 3. □

Proof : follows from the converse part of theorem 3 with $r.r'_n = s_0$. □

Proposition 4 : Let t_0 and t'_0 be two terms A-unifiable by a substitution s. Then there exists a R,E-narrowing derivation $t_0=t'_0 \overset{\wedge}{\longrightarrow}{}^{R,E} \dots \overset{\wedge}{\longrightarrow}{}^{R,E} t_n=t'_n$ such that (1) $\forall j$, $0 \le j \le n$, $r'_j{}_{|V(t_0) \cup V(t'_0)}$ is R,E-normalized.

(2) $\exists r$ in $CSU(t_n,t'_n,E)$, $r.r'_n$ A-unifies t_0 and t'_0 and $r.r'_n \le_A s$. □

Proof : follows from the direct part of theorem 3 where s_0 is obtained by R,E-normalizing s. □

Theorem 4 : Let A = RUE be an equational theory such that R is E-canonical and E-commuting. Let t_0 and t'_0 be two terms and S be the set of all substitutions s such that s is in S iff $t_0 = t'_0 \overset{\wedge}{\longrightarrow}{}^{R,E} \dots \overset{\wedge}{\longrightarrow}{}^{R,E} t_n=t'_n$ with t_n and t'_n E-unifiable, r'_n R,E-normalized and $s = r.r'_n$ where r is any substitution in a complete set of E-unifiers of t_n and t'_n away from $V(t_n) \cup V(t'_n)$. Then S is a complete set of A-unifiers of t_0 and t'_0 away from $V(t_0) \cup V(t'_0)$. □

5. ELIMINATION OF REDUNDANCIES AND TERMINATION CRITERION

Following Hullot, we now define the basic R,E-narrowing.

Definition 17 : a R,E-narrowing derivation starting in t :

$$t \overset{\wedge}{\longrightarrow}{}^{R,E}_{[u_0,i_0,r_0]} t_1 \dots \overset{\wedge}{\longrightarrow}{}^{R,E}_{[u_{j-1},i_{j-1},r_{j-1}]} t_j \overset{\wedge}{\longrightarrow}{}^{R,E} \dots$$

is basic iff for all j, u_j belongs to U_j where $U_0 = O(t)$ and

$$U_{j+1} = U_j \setminus \{u_j.v \mid u_j.v \in Dom(t_j)\} \cup \{u_j.v \mid v \in O(d_{r_j})\} .$$ □

Intuitively, in basic R,E-narrowing, no R,E-narrowing is applied at occurrences of subterms added by a previous R,E-narrowing substitution.

Example 3 : Let $R=\{(h(h(x)) \xrightarrow[1]{} h(x)$ and $h(a) \xrightarrow[2]{} a\}$ and $E=\{x+y=y+x\}$.

Then the following narrowing derivation is basic :

$$h(y)+y \overset{\wedge}{\longrightarrow}_{[1,1,\{y \leftarrow h(x)\}]} h(x)+h(x) \overset{\wedge}{\longrightarrow}_{[1,2,\{x \leftarrow a\}]} a+h(a)$$

since $U_0 = \{e,1\} = U_1$, but the following one is not :

$$h(y)+y \overset{\wedge}{\longrightarrow}_{[1,1,\{y \leftarrow h(x)\}]} h(x)+h(x) \overset{\wedge}{\longrightarrow}_{[2,2,\{x \leftarrow a\}]} h(a)+a.$$

Theorem 4 holds by using basic R,E-narrowing derivations only, because in the completeness proof of proposition 4, it is possible to choose an innermost-outermost computation of R,E-normal forms. Then the associated R,E-narrowing derivation is basic. We therefore get a sufficient criterion for termination :

Theorem 5 : Let $R = \{g_i \rightarrow d_i\}$ be a term rewriting system E-commuting, such that any basic R,E-narrowing derivation starting from any of the g_i's terminates. Then any R,E-narrowing derivation starting from any term terminates. □

Proof : It follows exactly Hullot's proof. [HUL,80] □

Proposition 5 : If the hypothesis of theorems 4 and 5 hold, and if complete sets of E-unifiers are finite, then the construction of theorem 4 leads to a complete A-unification algorithm. □

This last result allows to obtain complete and finite unification algorithms for the theories of example 1, assuming a complete and finite algorithm is known for their E theories. For examples 1.2, 1.3 and 1.4, E-unification can be solved [K&K,82] by an extension of the Martelli and Montanari method.

6. CONCLUSION

We have shown here a powerfull method to obtain complete algorithms for A-unification. It is incremental in the following sense : starting with a complete E-unification algorithm (possibly $E=\emptyset$), we can add new axioms, assuming they provide a set of rewrite rules R with the required good properties of theorem 4 and get a complete unification algorithm for the theory RUE by using the R,E-narrowing process.

Notice on the other hand that the method will allow the implementation of various extensions of the Knuth and Bendix completion procedure : for proving strong consistency of hierarchical abstract data types for instance, one can start from a set E of equations such that a finite and complete E-unification algorithm is known (may be $E=\emptyset$) and a set R of rules defining new operations; one can complete the set of rules in order to obtain E-confluence and E-commutation (and the consistency is proved if success is obtained) ; then one can start again to prove the consistency of the next enrichment, because a complete procedure for unification in the theory EUR is now available.

However, computing E-critical pairs is very hard by hand and forbids a practical use (for abelian group theory for instance) how long we don't have an implementation. It is our purpose now to implement the full method in the Formel system of G.Huet and his co-workers at Inria.

Acknowledgments : We thank Gerard Huet for helpfull discussions. This work was developped within the framework of EURECA project, supported by the GRECO programmation.

7. REFERENCES

[FAY,79] FAY M.: *First order unification in equational theory.* Proc. 4th Workshop on Automata Deduction Texas (1979).

[H&H,80] HUET G., HULLOT J.M.: *Proofs by induction in equational theories with constructors.* Proc. 21th Symp. on Foundation of Comp. Sc. (1980).

[H&O,80] HUET G., OPPEN D.C.: *Equations and rewrite rules: a survey.* in *Formal Languages: Perspectives and open problems.* Ed. Book R., Academic Press (1980).

[HUE,77&80] HUET G.: *Confluent reductions: abstract properties and applications to term rewriting systems.* 18th. FOCS (1977) & J.ACM 27-4, pp. 797-821 (1980).

[HUL,80] HULLOT J.M.: *Canonical forms and unification.* Proc. 5th Workshop on Automated Deduction, Les Arcs (1980).

[K&B,70] KNUTH D., BENDIX P.: *Simple word problems in universal algebras.* in *Computational problems in abstract algebra.* Leech J. ed. Pergamon Pr. (1970).

[KKJ,81] KIRCHNER C., KIRCHNER H., JOUANNAUD J.P.: *Algebraic manipulations as a unification and matching strategy for linear equations in signed binary trees.* Proc. IJCAI 81, Vancouver (1981).

[K&K,82] KIRCHNER C., KIRCHNER H.: *Résolution d'équations dans les algèbres libres et les variétés équationnelles d'algèbres.* Thèse de 3e cycle, Université de Nancy (1982).

[LAN,79] LANKFORD D.S.: *Canonical inference.* Report ATP-32, Dpt. Math. and Comp. Sc., Univ. of Texas at Austin (1975).

[LAN,79] LANKFORD D.S.: *A unification algorithm for abelian group theory.* Report MTP-1, Math. dep., Louisiana Tech. U. (1979).

[L&S,77] LIVESEY M., SIEKMANN J.: *Unification of sets.* Report 3/76, Institut fur Informatik 1, Univ. Karlsruhe (1977).

[M&M,79] MARTELLI A., MONTANARI U.: *An efficient unification algorithm.* Technical report, University of Pisa (1979).

[P&S,81] PETERSON G.E., STICKEL M.E.: *Complete sets of reductions for equational theories with complete unification algorithms.* J. ACM 28, n°2, pp.233-264 (1981).

[PLO,72] PLOTKING G.: *Building in equational theories.* Machine Intelligence 7, pp.73-90 (1972).

[R&S,78] RAULEFS P., SIEKMANN J.: *Unification of idempotent functions.* Report, Institut fur Informatik 1, Univ. Karlsruhe (1978).

[STI,81] STICKEL M.E.: *A unification algorithm for associative-commutative functions.* J. ACM 28, n°3, pp.423-434 (1981).

CHARACTERIZATION OF WEIGHT VECTORS FOR WEIGHTED SYNCHRONIC DISTANCE IN PETRI NETS

ICHIRO SUZUKI
Education Center for
Information Processing
Osaka University
Toyonaka 560
JAPAN

TADAO KASAMI
Faculty of Engineering Science
Osaka University
Toyonaka 560
JAPAN

ABSTRACT

Three different notions of weighted synchronic distance in Petri nets are studied. We present (1) algorithms for finding a basis of the linear space of weight vectors that yield a finite distance, (2) a method for computing the distance for any given weight vector by examining a finite set of vectors obtained a priori, and (3) some complexity results.

1. INTRODUCTION

Synchronic distance in Petri nets [8,14] is a measure of synchronic dependence between transition firings first introduced by Petri [15]. The notion is generalized to weighted synchronic distance in [3,5]. Other variations are found in [1,13].

The weighted synchronic distance we study in this paper is a natural generalization of the notions in [3,5]---we consider the synchronic dependence of all transition firings by assigning weights to all transitions. Therefore a weight is represented by a vector over integers, called a weight vector, whose dimension is the number of transitions. The weighted synchronic distance with respect to a weight vector y is called the y-distance. The transitions are y-synchronized if the y-distance is finite. For a Petri net N, the set SYNC(N) of weight vectors y for which the transitions are y-synchronized is a linear space. As one of the main results, we present an algorithm for obtaining a basis of SYNC(N) for arbitrary Petri net N. Our result is more general than those found in [3,5]---[3] presents a method for obtaining a weight that yields a finite distance for "simple place/transition nets called regulation circuits", and [5] characterizes weights that yield a finite distance for "cyclic, contact-free, finite condition/event systems" (so the set of reachable markings is necessarily finite in our terminology). Furthermore, we present a method for computing the y-distance for any given weight vector y in SYNC(N) by examining a finite set of vectors that can be obtained a priori. Also it is shown that the membership problem for SYNC(N) is exponential-space-hard.

Next, we introduce two measures of synchronic dependence between transition firings, the complete y-dependence and the potential y-dependence. Given a Petri net N with a set of transitions T and an initial marking M_0, a subset $T_1 \subseteq T$ of transitions, and a weight vector y, the transitions in $T-T_1$ are completely y-dependent (or potentially y-dependent) on T_1 if the y-distance starting from M without firing the transitions in T_1 is finite for any (or some) marking M reachable from M_0. (The idea of measuring the dependence by the number of firings of one transition without firing another has already been proposed in [13].) $CDEP(N,T_1)$ (or $PDEP(N,T_1)$) denotes the set of weight vectors y for which the transitions in $T-T_1$ are completely y-dependent (or potentially y-dependent) on T_1. $CDEP(N,T_1)$ is a linear space, and we present a method for obtaining a basis of $CDEP(N,T_1)$ for arbitrary Petri net N. The membership problem for CDEP(N,T_1) is shown to be exponential-space-hard. As for the potential dependence, it is shown that the membership problem for PDEP(N,T_1) is equivalent to the reachability problem for Petri nets, and thus is solvable but computationally intractable in general.

Finally, a dual notion of the y-synchronization for token counts of markings is studied.

2. DEFINITIONS

$\mathbb{Z}$ and $\mathbb{N}$ denote the set of integers and the set of nonnegagive integers, respectively. For a set S and $n \in \mathbb{N}$, S^n represents the set of $n \times 1$ column vectors over S. The i-th component of a vector v is denoted by v_i. v^T is the transpose of a vector or a matrix v. $u \geq v$ is the componentwise comparison of vectors u and v. We write $u > v$ if $u \geq v$ and $u \nleq v$. $u \cdot v$ is the usual multiplication of scalars, vectors or matrices u and v. For a set S, S^* denotes the set of all strings composed of elements of S including the empty string λ. We use "$\cdot$" for representing both the concatenation of strings and the usual multiplication stated above. However no confusion will arise from the context.

A _Petri net_ is a quintuple $N=(P,T,F,B,M_0)$ where:

(a) $P=\{p_1,p_2,\ldots,p_m\}$ is a finite set of _places_.

(b) $T=\{t_1,t_2,\ldots,t_n\}$ is a finite set of _transitions_ such that $P \cap T = \phi$ (ϕ denotes the empty set).

(c) $F: T \to \mathbb{N}^m$ is a _forward incidence function_.

(d) $B: T \to \mathbb{N}^m$ is a _backward incidence function_.

(e) $M_0 \in \mathbb{N}^m$ is an _initial marking_.

Any vector $M \in \mathbb{N}^m$ is called a _marking_. For each i, $1 \leq i \leq m$, place p_i is said to hold M_i tokens at marking M. The $n \times m$ matrix $A=[B(t_1)-F(t_1),\ldots,B(t_n)-F(t_n)]^T$ is called the _incidence matrix_ of N. For $\alpha \in T^*$, $\theta(\alpha) \in \mathbb{N}^n$ is the _Parikh mapping_ [4] of α whose i-th component $\theta(\alpha)_i$ is the number of transition t_i's occurrences in α. A transition is _firable_ at marking M if and only if $M \geq F(t)$. For a marking $M \in \mathbb{N}^m$, the set $L(N,M) \subseteq T^*$ of _firing sequences_ is the smallest set obtained recursively by:

(a) $\lambda \in L(N,M)$.

(b) For $\alpha \in L(N,M)$ and $t \in T$, $\alpha \cdot t \in L(N,M)$ if $M + A^T \cdot \theta(\alpha) \geq F(t)$.

For firing sequence $\alpha \in L(N,M)$, we say that marking $M' = M + A^T \cdot \theta(\alpha)$ is _reachable_ from M by α, and write $M \xrightarrow[N]{\alpha} M'$. $R(N,M) = \{M' \mid M \xrightarrow[N]{\alpha} M'$ for some $\alpha \in L(N,M)\}$ is the set of markings reachable from M. $L(N,M_0)$ and $R(N,M_0)$ are simply written as $L(N)$ and $R(N)$, respectively, where M_0 is the initial marking.

3. Y-SYNCHRONIZATION

Let $N=(P,T,F,B,M_0)$ be a Petri net with $|T|=n$. For a weight vector

$y \in \mathbb{Z}^n$, we say that the transitions are *y-synchronized* if *y-distance* $\max_{\alpha \in L(N)} (|y^T \cdot \theta(\alpha)|) < \infty$. We let

$$SYNC(N) = \{y \in \mathbb{Z}^n | \max_{\alpha \in L(N)} (|y^T \cdot \theta(\alpha)|) < \infty\}$$

denote the set of weight vectors y for which the transitions are y-synchronized. Clearly SYNC(N) is a linear space. In this section we will show that a basis of SYNC(N) can be obtained effectively, and present a method for computing y-distance for any weight vector $y \in SYNC(N)$.

For a Petri net $N=(P,T,F,B,M_0)$ where $|P|=m$ and $|T|=n$, we define the coverability graph G(N) of N which is essentially the tree of Karp and Miller [8] with some additional arcs (similar to the backpointers in the coverability tree of Hack [7]). G(N) is a labelled directed graph with two kinds of arcs, *forward arcs* and *backward arcs*. We write $v \overset{e}{\Longrightarrow} v'$ if there is an arc (either a forward arc or a backward arc) e from node v to v' in G(N). Suppose that $v_1 \overset{e_1}{\Longrightarrow} v_2$, $v_2 \overset{e_2}{\Longrightarrow} v_3, \ldots,$ and $v_k \overset{e_k}{\Longrightarrow} v_{k+1}$ for some $k \geq 1$, and let $p = e_1 \cdot e_2 \cdot \ldots \cdot e_k$. Then p is called a *path* from node v_1 to v_{k+1}, and we write $v_1 \overset{p}{\Longrightarrow} v_{k+1}$. Path p is an *elementary path* if $v_1, v_2, \ldots,$ and v_{k+1} are all distinct. Path p is called a *loop* if $v_1 = v_{k+1}$. Loop p is an *elementary loop* if $v_1 (=v_{k+1}), v_2, \ldots,$ and v_k are all distinct. For convenience we may write $v \overset{\lambda}{\Longrightarrow} v$ for node v. The label of a node v, denoted by $\ell(v)$, is an element of $(\mathbb{N} \cup \{\omega\})^m$ where ω is a symbol such that $n < \omega$ and $n + \omega = \omega$ for any $n \in \mathbb{Z}$. The label of an arc e is an element of $T \cup \{\lambda\}$, and is denoted by $\ell(e)$. For a path or a loop $p = e_1 \cdot e_2 \cdot \ldots \cdot e_k$, we let $\ell(p) = \ell(e_1) \cdot \ell(e_2) \cdot \ldots \cdot \ell(e_k)$ denote the sequence of labels of arcs in p.

Formally, the *coverability graph* G(N) of a Petri net $N=(P,T,F,B,M_0)$ is constructed recursively by the following rules:

Step(1) Create an unmarked node v_0 with $\ell(v_0) = M_0$ ("marked" should not be confused with "labelled").

Step(2) Let v be an unmarked node. Do the following operation for each $t \in T$ such that $\ell(v) \geq F(t)$, and after that mark v (let $M' = \ell(v) - F(t) + B(t)$):

(a) If there exists a node v' (possibly equal to v) on the elementary path from v_0 to v such that $M' = \ell(v')$, then create a backward arc labelled t from v to v'.

(b) If no such v' cited in (a) exists, then create an unmarked node v_1 and a forward arc labelled t from v to v_1. For each i, $1 \leq i \leq m$, $\ell(v_1)_i = \omega$ if $M' > \ell(v'')$ and $M'_i > \ell(v'')_i$ for some v'' (possibly equal to v) on the elementary path from v_0 to v, and $\ell(v_1)_i = M'_i$ otherwise. Also, create a backward arc labelled λ from v_1 to each v'' (possibly equal to v) on the elementary path from v_0 to v such that $M' > \ell(v'')$.

Step(3) Repeat Step(2) until there remains no unmarked node.

The construction of G(N) terminates in a finite number of steps by an argument similar to the one in [8] based on König's theorem [9].

An execution of Step(2) during the construction of G(N) is called a loop-introducing step if at least one backward arc is created in it. The rank of a backward arc that is added to the graph during the k-th loop-introducing step is defined to be k. The rank of an elementary loop is defined to be the rank of the associated backward arc.

Lemmas 1 and 2 are well-known [7,8].

Lemma 1 For each node v of G(N) and any marking $M \in \mathbb{N}^m$ such that $\ell(v) \geq M$, there exists a firing sequence $\alpha \in L(N)$ such that,

(a) $M_0 \xrightarrow[N]{\alpha} M'$,

(b) $M' \geq M$, and

(c) there exists a path p from v_0 to v in G(N) such that (i) $\ell(p)=\alpha$, (ii) any node on p lies on the elementary path from v_0 to v, and (iii) every backward arc e on p has $\ell(e)=\lambda$. $\square$

Lemma 2 For any firing sequence $\alpha \in L(N)$, there exists a path p from v_0 in G(N) such that (i) $\ell(p)=\alpha$ and (ii) no backward arc e on p has $\ell(e)=\lambda$. $\square$

Lemma 3 Let $N=(P,T,F,B,M_0)$ be a Petri net with $|T|=n$. For any $y \in \mathbb{Z}^n$, the following statements are equivalent:

(a) $y \in SYNC(N)$.

(b) $y^T \cdot \theta(\ell(p))=0$ for every elementary loop p of G(N) whose associated backward arc has label other than λ.

(Proof) (a)$\Longrightarrow$(b): Let $|P|=m$. We prove by induction on the rank of elementary loops in G(N).

(basis) Let e be a backward arc with rank 1. Suppose that $v_0 \xRightarrow{p} v$, $v \xRightarrow{p'} v'$ and $v' \xRightarrow{e} v$ where $p \cdot p'$ is the elementary path from v_0 to v'. Note that $p'=\lambda$ if $v=v'$, and that $p' \cdot e$ is an elementary loop with rank 1. By definition of G(N), we have $\ell(p) \cdot (\ell(p' \cdot e))^k \in L(N)$ for any $k \in \mathbb{N}$. Since

$$y^T \cdot \theta(\ell(p) \cdot (\ell(p' \cdot e))^k) = y^T \cdot \theta(\ell(p)) + k \cdot y^T \cdot \theta(\ell(p' \cdot e))$$

for any large k, $y \in SYNC(N)$ only if $y^T \cdot \theta(\ell(p' \cdot e))=0$.

(induction) Assume that $y^T \cdot \theta(\ell(p))=0$ for all elementary loops p with rank less than j ($j \geq 2$). Let e be a backward arc with rank j. We have two cases.

(Case 1) $\ell(e) \neq \lambda$: Suppose that $v_0 \xRightarrow{p} v$, $v \xRightarrow{p'} v'$ and $v' \xRightarrow{e} v$ where $p \cdot p'$ is the elementary path from v_0 to v'. Note that $p'=\lambda$ if $v=v'$, and that $p' \cdot e$

is an elementary loop with rank j. For each $k\in\mathbb{N}$, let $M_k\in\mathbb{Z}^m$ be the marking defined as follows: For each i, $1\le i\le m$,

$$(M_k)_i = \begin{cases} \ell(v)_i & \text{if } \ell(v)_i\neq\omega \\ \rho((p'\cdot e)^k)_i & \text{if } \ell(v)_i=\omega \end{cases}$$

where $\rho((p'\cdot e)^k)=[F(t_1),\ldots,F(t_n)]\cdot\theta(\ell((p'\cdot e)^k))$. Then it is easy to see that

$$(\ell(p'\cdot e))^k\in L(N,M_k). \tag{1}$$

Since $\ell(v)\ge M_k$, by Lemma 1 there exists a firing sequence $\alpha_k\in L(N)$ such that:

(a) $M_0 \overset{\alpha_k}{\underset{N}{\dashrightarrow}} M$.

(b) $M\ge M_k$.

(c) $\alpha_k=\ell(p_k)$ for some path p_k in G(N) from v_0 to v such that every node on p_k is also on the elementary path p from v_0 to v.

(d) Every backward arc on p_k has label λ.

So $\alpha_k\cdot(\ell(p'\cdot e))^k\in L(N)$ by (a), (b) and (1). Any elementary loop embedded in p_k has rank less than j by (c) and (d), and thus by the inductive hypothesis $y^T\cdot\theta(\ell(p_k))=y^T\cdot\theta(\ell(p))$. So

$$y^T\cdot\theta(\alpha_k\cdot(\ell(p'\cdot e))^k)=y^T\cdot\theta(\ell(p))+k\cdot y^T\cdot\theta(\ell(p'\cdot e))$$

for any large k, and therefore $y\in SYNC(N)$ only if $y^T\cdot\theta(\ell(p'\cdot e))=0$.

(Case 2) $\ell(e)=\lambda$: Suppose that $v_0\overset{p}{\Longrightarrow}v$, $v\overset{p'}{\Longrightarrow}v'$ and $v'\overset{e}{\Longrightarrow}v$ where $p\cdot p'$ is the elementary path from v_0 to v'. Note that $e\cdot p'$ is an elementary loop with rank j. Let v" (possibly equal to v) be the node such that $v''\overset{e'}{\Longrightarrow}v'$ where e' is a forward arc on p', and let $p'=p''\cdot e'$. Note that $p''=\lambda$ if $v''=v$. For each $k\in\mathbb{N}$, let $M_k\in\mathbb{Z}^m$ be the marking defined as follows: For each i, $1\le i\le m$,

$$(M_k)_i = \begin{cases} \ell(v'')_i & \text{if } \ell(v'')_i\neq\omega \\ \rho(e'\cdot(e\cdot p')^k)_i & \text{if } \ell(v'')_i=\omega \end{cases}$$

where $\rho(e'\cdot(e\cdot p')^k)=[F(t_1),\ldots,F(t_n)]\cdot\theta(\ell(e'\cdot(e\cdot p')^k))$. Then it is easy to see that

$$\ell(e'\cdot(e\cdot p')^k)\in L(N,M_k). \tag{2}$$

Since $\ell(v'')\ge M_k$, by Lemma 1 there exists a firing sequence $\alpha_k\in L(N)$ such that:

(a) $M_0 \overset{\alpha_k}{\underset{N}{\dashrightarrow}} M$.

(b) $M\ge M_k$.

(c) $\alpha_k=\ell(p_k)$ for some path p_k in G(N) from v_0 to v" such that every node on p_k is also on the elementary path $p\cdot p''$ from v_0 to v".

(d) Every backward arc on p_k has label λ.

So $\alpha_k\cdot\ell(e'\cdot(e\cdot p')^k)\in L(N)$ by (a), (b) and (2). Any elementary loop embedded in p_k has rank less than j by (c) and (d), and thus by the inductive hypothesis $y^T\cdot\theta(\ell(p_k))=y^T\cdot\theta(\ell(p\cdot p''))$. So

$$y^T \cdot \theta(\alpha_k \cdot \ell(e' \cdot (e \cdot p')^k)) = y^T \cdot \theta(\ell(p \cdot p'' \cdot e')) + k \cdot y^T \cdot \theta(\ell(e \cdot p'))$$

for any large k, and therefore $y \in SYNC(N)$ only if $y^T \cdot \theta(\ell(e \cdot p')) = 0$.

(b)===>(a): This direction is straightforward by Lemma 2 and the finiteness of G(N). □

By Lemma 3 and a standard linear algebraic method [11], we have:

Theorem 1 For any Petri net N, a basis of SYNC(N) can be obtained effectively. □

Theorem 2 Let $b_1, \ldots, b_K \in \mathbb{Z}^n$ be a basis of SYNC(N). We can obtain a finite collection $u_1, \ldots, u_Q \in \mathbb{Z}^K$ of vectors, where Q is no larger than the number of nodes in G(N), such that for weight vector $y = [b_1, \ldots, b_K] \cdot c \in SYNC(N)$ where $c \in \mathbb{Z}^K$ we have $\max_{\alpha \in L(N)} (|y^T \cdot \theta(\alpha)|) = \max_{1 \le i \le Q} (|c^T \cdot u_i|)$.

(Proof) Omitted due to space limitation. □

Since the construction of G(N) requires time exponential to the size of N in the worst case, a faster algorithm is desired. However the following theorem shows that no drastic improvement can be found. The proof is by reduction of the boundedness problem for Petri nets, which is exponential-space-hard [10], to the membership problem for SYNC(N).

Theorem 3 The problem of deciding whether $y \in SYNC(N)$ for arbitrary Petri net N and weight vector y is exponential-space-hard. □

4. COMPLETE Y-DEPENDENCE AND POTENTIAL Y-DEPENDENCE

In this section we investigate two new measures for synchronic dependence of transition firings.

Consider Petri net N_1 shown in Fig. 1 in the usual graphical representation [14]. It is easy to see that $SYNC(N_1) = \{(0,0)^T\}$, and thus we should rather say that t_1 and t_2 are "asynchronous". However the firings of t_2 is dependent on those of t_1 in the sense that at any marking t_2 will eventually be unable to fire if t_1 ceases to fire. We propose new measures for dealing with such situations.

Let $N = (P, T, F, B, M_0)$ be a Petri net with $|P| = m$ and $|T| = n$. Let $T_1 \subseteq T$ be a subset of transitions. For a marking $M \in \mathbb{N}^m$ and a weight vector $y \in \mathbb{N}^n$, let

$$\Sigma_{M,y}(N, T_1) = \max_{\substack{\alpha \in L(N,M) \text{ with} \\ \theta(\alpha)_i = 0 \text{ for all } t_i \in T_1}} (|y^T \cdot \theta(\alpha)|).$$

$\Sigma_{M,y}(N, T_1)$ is the y-distance starting from M without firing the transitions in T_1. We say that $T-T_1$ is completely y-dependent (or potentially y-dependent) on T_1 if $\Sigma_{M,y}(N,T_1)<\infty$ for any (or some) marking $M\in R(M_0)$. Then in N_1 of Fig. 5, $\{t_2\}$ is completely y-dependent on $\{t_1\}$ for any weight vector y (so t_2 is "synchronized" with t_1 as our intuition says).

Let us define the following sets of weight vectors:

$CDEP(N,T_1) = \{y\in\mathbb{Z}^n \mid \Sigma_{M,y}(T_1)<\infty$ for any $M\in R(M_0)\}$

$PDEP(N,T_1) = \{y\in\mathbb{Z}^n \mid \Sigma_{M,y}(T_1)<\infty$ for some $M\in R(M_0)\}$

Clearly $CDEP(N,T_1)$ is a linear space.

Theorem 4 For any Petri net $N=(P,T,F,B,M_0)$ and a subset $T_1\subseteq T$ of transitions, a basis of $CDEP(N,T_1)$ can be obtained effectively.

(Proof) Let $|P|=m$. Let $G'(N)$ be the graph obtained from the coverability graph $G(N)$ by removing all arcs labelled by either λ or any $t_i\in T_1$. For each node v of $G'(N)$, let $Reach_v$ be the set of nodes reachable from v by a path of length zero or more. Let V be the set of nodes v of $G'(N)$ such that there is no node $v'\in Reach_v$ with $\ell(v)<\ell(v')$. For each $v\in V$, let $W_v=\{i \mid 1\le i\le m, \ell(v)_i=\omega\}$. Let $Loop_v=\{\theta(\ell(p)) \mid p$ is a loop in $G'(N)$ containing $v\}$. By Parikh's theorem [4] $Loop_v$ is a computable [6] semilinear set. For two disjoint subsets I, $J\subseteq\{1,2,...,m\}$ of indices, let $S_{I,J}=\{x\ge 0_{m\times 1} \mid (A^T\cdot x)_i\ge 1$ for $i\in I$, $(A^T\cdot x)_i\ge 0$ for $i\in J\}$. $S_{I,J}$ is a computable semilinear set [6]. For $v\in V$ and $Y\subseteq\{1,2,...,m\}$ such that $Loop_v\cap S_{\phi,Y}\neq\phi$, there is a subset I of Y such that:

(a) $Loop_v\cap S_{Y-I,I}\neq\phi$.

(b) For any $I'\subseteq Y$ with $Loop_v\cap S_{Y-I',I'}\neq\phi$, $I\subseteq I'$ holds.

Such I will be denoted $W(v,Y)$. We have:

(P1): Suppose that $Loop_v\cap S_{\phi,Y}\neq\phi$, i.e., $W(v,Y)$ is defined. By the definition of $W(v,Y)$, for $Y'\subseteq Y$, $W(v,Y')$ is defined and $W(v,Y')\subseteq W(v,Y)$.

Consider the following algorithm.

Step(1) $L_v:=\{W_v\}$.

Step(2) For each pair of $v\in V$ and $X\in L_v$ such that $W(v,X)$ is defined, do the following: For each $v'\in Reach_v$, let $L_{v'}:=L_{v'}\cup W(v,X)$, and then remove all $x\in L_{v'}$ such that there is another $x'\in L_{v'}$ with $x'\subset x$ (we write $x'\subset x$ if $x'\subseteq x$ and $x'\neq x$).

Step(3) Repeat Step(2) until no more application is possible.

Step(4) $CDEP^{\perp}:= \bigcup_{v\in V,\ X\in L_v} (Loop_v\cap S_{\phi,X})$.

(P2): Suppose that (1) there is $Y_1\subseteq Y$ in L_v, (2) v' is reachable from v, and (3) $W(v',Y)$ is defined. Then by (P1) $W(v',Y_1)$ is defined, and by Step(2) there must exist Y' in $L_{v'}$ such that $Y'\subseteq W(v',Y_1)\subseteq W(v',Y)$.

Note that all set operations above on semilinear sets can be carried out effectively [6]. Now we show that $y\in CDEP(N,T_1)$ if and only if $y^T\cdot x=0$ for all $x\in CDEP^{\perp}$. Then a basis of $CDEP(N,T_1)$ can be obtained easily from the generators [6] of semilinear set $CDEP^{\perp}$ obtained effectively by the algorithm.

(if) For $v'\in V$, let ℓ be a label obtained from $\ell(v')$ by replacing the ω-components of $\ell(v')$ by some arbitrary nonnegative integers. Consider the process of constructing the coverability graph starting from a node v with label ℓ.

Suppose that we obtain loops p_1 containing node v_1, p_2 containing node $v_2,\ldots$, and p_k containing node v_k such that (let $v_0=v$):

(a) v_{j+1} is reachable from v_j, $0\le j<k$.

(b) $X_0=W_{v_1'}$.

(c) $(A^T\cdot\theta(\ell(p_j)))_i=0$, $i\in X_j$, and
$(A^T\cdot\theta(\ell(p_j)))_i\ge 1$, $i\in X_{j-1}-X_j$, $1\le j<k$.

(d) $(A^T\cdot\theta(\ell(p_k)))_i\ge 0$, $i\in X_k$.

(e) $X_1\supset X_2\supset\ldots\supset X_k$.

By the definition of $W(v,Y)$ and (c), $W(v_j',X_{j-1})$ is defined and

$$W(v_j',X_{j-1})\subseteq X_j,\ 1\le j\le k. \tag{*1}$$

By Step(2) there is X_1' in $L_{v_1'}$ such that

$$X_1'\subseteq W(v_1',M_{v_1'})\subseteq X_1. \tag{*2}$$

It follows from (*1), (*2) and (P2) that there is X_j' in $L_{v_j'}$ with $1\le j\le k$ such that

$$X_j'\subseteq W(v_j',X_{j-1})\subseteq X_j.$$

Since $(A^T\cdot\theta(\ell(p_k)))_i\ge 0$ for $i\in X_k$, $X_k'\subseteq X_k$ and $X_k'\in L_{v_k'}$, we have $\theta(\ell(p_k))\in CDEP^{\perp}$.

(only if) Let $x\in CDEP^{\perp}$. By Step(4) there are $v\in V$, $X\in L_v$ and a loop p containing v such that $\theta(\ell(p))=x$ and $(A^T\cdot x)_i\ge 0$ for $i\in X$.

(Case 1) $X=M_v$: If we have "enough" tokens on all places corresponding to the ω-components of the label $\ell(v)$, the firing sequence associated with loop p can be repeated as many times as desired, and thus we must have $y^T\cdot x=0$ for any $y\in CDEP(N,T_1)$.

(Case 2) $X\neq M_v$: By Step(2) there exist nodes $v_1, v_2,\ldots,v_{k-1},v_k=v$, and sets of indices $X_1=W_{v_1}, X_2,\ldots, X_k=X$ such that:

(a) v_{j+1} is reachable from v_j, $1\le j<k$.

(b) $W(v_j,X_j)$ is defined, $1\le j<k$.

(c) $X_{j+1}=W(v_j,X_j)$, $1\le j<k$.

Then there exist loops p_j containing node v_j, $1\le j<k$, such that

$(A^T \cdot \theta(\ell(p_j)))_i \geq 1$, $i \in X_j - X_{j+1}$, and

$(A^T \cdot \theta(\ell(p_j)))_i \geq 0$, $i \in X_{j+1}$, $1 \leq j < k$.

Let $p_k = p$. If we have "enough" tokens on the ω-components of the marking (label) associated with node v_1, each of the firing sequences with respect to loops $p_1, p_2, \ldots,$ and p_k can be repeated as many times as desired, and thus by a similar argument required for deriving Lemma 3 we can show that $y^T \cdot \theta(\ell(p_j)) = 0$, $1 \leq j \leq k$, must hold for any $y \in CDEP(N,T_1)$. ⊓

By definition $y \in SYNC(N)$ if and only if $y \in CDEP(N,\phi)$. So the problem of deciding whether $y \in CDEP(N,T_1)$ for arbitrary Petri net N, subset T_1 of transitions, and weight vector y is exponential- space-hard.

Theorem 5 The problem (MEM-PDEP) of deciding whether $y \in PDEP(N,T_1)$ for arbitrary Petri net N, a subset T_1 of transitions and a weight vector y is equivanent to the reachability problem.

(Proof) (I) Here we show that MEM-PDEP is no easier than the single transition liveness problem which is equivalent [7] to the reachability problem. A transition t of a Petri net N is live if for any marking $M \in R(N)$, $\alpha \cdot t \in L(N,M)$ holds for some firing sequence α. The single transition liveness problem is to decide whether a particular transition of a given Petri net is live.

Suppose that we wish to test whether transition $t_1 \in T$ of a Petri net $N=(P,T,F,B,M_0)$ is live or not. Let $y=(1,0,\ldots,0)^T \in \mathbb{Z}^n$ where $|T|=n$. By difinition t_1 is live if and only if $\Sigma_{M,y}(N,\phi)=\infty$ for any $M \in R(N)$. Thus we can test the liveness of t_1 by checking if $y \in PDEP(N,\phi)$ or not.

(II) Next we show that MEM-PDEP is no more difficult than the reachability problem. Suppose that we wish to test whether $y \in PDEP(N,T_1)$ or not for a given Petri net $N=(P,T,F,B,M_0)$, a subset $T_1 \subseteq T$ of transitions, and a weight vector $y \in \mathbb{Z}^n$ where $|T|=n$. Since $PDEP(N,T_1)=\mathbb{Z}^n$ if $T_1=T$, in the following we assume that T_1 is a proper subset of T. Without loss of generality let $T-T_1=\{t_1,t_2,\ldots,t_{n_1}\}$ for some n_1, $1 \leq n_1 \leq n$. Note that by definition, for any M, $\{y \in \mathbb{Z}^n | \Sigma_{M,y}(N,T_1)<\infty\} = SYNC(N') \times \mathbb{Z}^{(n-n_1)}$ where "×" denotes the direct product, and N' is the Petri net obtained from N by removing all transitions in T_1 and replacing the initial marking M_0 by M. So by Theorem 1 we can test whether $\Sigma_{M,y}(N,T_1)<\infty$ or not. It is known [2] that if we have an algorithm to solve the reachability problem, then we can obtain Min(R(N)) effectively where Min(R(N)) is the smallest subset of R(N) such that, for any $M \in R(N)$ there exists a marking $M' \in Min(R(N))$ with $M \geq M'$. Since $\Sigma_{M,y}(N,T_1)<\infty$ for some $M \in R(N)$ if and only if $\Sigma_{M',y}(N,T_1)<\infty$ for some

$M' \in Min(R(N))$, and $Min(R(N))$ is a finite set by König's theorem [9], it is decidable whether $y \in PDEP(N,T_1)$ or not. ∏

Since the reachability problem is solvable [12] and exponential-space-hard [10], MEM-PDEP is solvable, but is exponential-space-hard.

5. Y-SYNCHRONIZATION FOR TOKEN COUNTS

For Petri net $N=(P,T,F,B,M_0)$ with $|P|=m$, a basis of the linear space $SYNC_P(N) = \{y \in \mathbb{Z}^m \mid \max_{M \in R(N)} (|y^T \cdot M|) < \infty\}$ can be obtained effectively.

REFERENCES

1. André, C., Armand, P., Boeri, F.: Synchronic relations and applications in parallel computation. Digital Processes 5, 99-113 (1979).
2. Araki. T, Kasami, T.: Some decision problems related to the reachability problem for Petri nets. Theoretical Computer Science 3, 85-104 (1977).
3. Genrich, H.J., Lautenbach, K., Thiagarajan, P.S.: Elements of general net theory. In: Net Theory and Applications, Lecture Notes in Computer Science 84, Springer-Verlag 1980.
4. Ginsburg, S.: The Mathematical Theory of Context Free Languages. McGraw Hill 1966.
5. Goltz, U., Reisig, W.: Weighted synchronic distance. In: Applications and Theory of Petri nets, Informatik-Fachberichte, pp. 289-300, Springer-Verlag 1982.
6. Grabowski, J.: Linear methods in the theory of vector addition systems I. Elektronische Informationsverarbeitung und Kybernetik 16, 207-236 (1980).
7. Hack, M.: Decision problems for Petri nets and vector addition systems. Computation Structure Group Memo 95-1, Project MAC, Massachusetts Institute of Thchnology, Cambridge, Massachusetts (1974).
8. Karp, R., Miller, R.: Parallel Program Schemata. Journal of Computer and System Sciences 3, 147-195 (1969).
9. König, D: Theorie der Endlichen und Unendlichen Graphen, Akademische Verlagsgesellschaft, Leipzig 1936.
10. Lipton, R.: The reachability problem and the boundedness problem for Petri nets are exponential-space hard. In: Conference on Petri Nets and Related Methods, M.I.T., 1975.
11. MacLane, S., Birkhoff, G.: Algebra. Macmillian, New York 1967.
12. Mayr, E.W.: An algorithm for the general Petri net reachability problem. Proceedings of the 13th Annual ACM Symposium on Theory of Computing, 238-246 (1981).
13. Murata, T., Le, V.B., Leu, D.J.: A method for realizing the synchronic distance matrix as a marked graph. Proceedings of the 1982 IEEE International Symposium on Circuits and Systems (1982).
14. Peterson, J.L.: Petri Net Theory and the Modelling of Systems. Prentice-Hall 1981.
15. Petri, C.A.: Interpretation of net theory. Interner Bericht 75-07, ISF-GMD (1975).

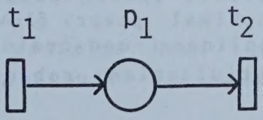


Fig. 1 Petri net N_1.

LARGE SCALE NONLINEAR PROGRAMMING

LEON S. LASDON
Graduate School of Business
University of Texas
Austin, Texas 78712

ALLAN D. WAREN
Computer Science Dept.
Cleveland State University
Cleveland, Ohio 44115

ABSTRACT

Our ability to solve large nonlinear programs has increased significantly over the last several years. Some new algorithms have been developed, existing ones have been refined, some good software has become available , much new computational experience has been reported and a number of practical applications described. This paper summarizes some of this activity, focusing on Successive Linear Programming (SLP), Successive Quadratic Programming (SQP), and Generalized Reduced Gradient (GRG) algorithms.

Scope

The field of nonlinear programming has been characterized by a wealth of theory and algorithms, but only a few good software packages and little knowledge of comparative algorithm performance. Consequently applications have lagged behind our modelling and solution capabilities. Over the last several years this situation has changed dramatically. Good software is now available. Recent computational testing has clarified the relative efficiencies of various algorithms, as well as identifying good codes. Predictably, applications are increasing, but much remains to be done in this arena.

Part of this "applications gap" is due to the lack of information, available to practicing engineers, concerning the capabilities and potential benefits of Nonlinear Programming. We attempt to partially fill this information void by summarizing some recent developments.

The paper begins by briefly describing the three most promising classes of NLP algorithms: generalized reduced gradient (GRG), successive linear programming (SLP), and successive quadratic programming (SQP). It then focuses on large nonlinear programs, provides three alternative forms for such large problems, and describes problem types which lead to such forms. The final section summarizes the relative advantages and disadvantages of these three classes of methods.

This paper draws on some material from two earlier survey papers, one on NLP algorithms and software [Waren and Lasdon, 1979], the other on NLP applications [Lasdon and Waren, 1980].

Conclusions and Significance

There now exists enough experience to indicate that solving large nonlinear optimization problems of certain types is both feasible and practical. These include problems with only linear constraints, optimal power flow problems, and problems which may have nonlinear constraints but are "mostly" linear. Solving large optimization problems is still far from routine, but several algorithms exist which show considerable promise.

Given the wide variety of nonlinear problems, it is unlikely that there exist a globally best nonlinear optimization algorithm or code. A sensible approach is to build a small library of well implemented, easy-to-use codes from the classes of algorithms suggested here. Although special implementations are required for large problems, some are now appearing. More experience is needed to assess the relative merits of these approaches, both in general and for specific applications.

1. Introduction

This survey briefly describes algorithms and problem formulations for large nonlinear optimization problems. Some existing software is also discussed. A general form for such problems is:

Problem NLP

minimize f(x)

subject to

$$\ell_i \leq g_i(x) \leq u_i \qquad i = 1, m \qquad (1)$$

$$\ell_{m+i} \leq x_i \leq u_{m+i} \qquad i = 1, n \qquad (2)$$

In the above, x is an n-component vector with components x_i, and ℓ_i, u_i are specified lower and upper bounds. If $\ell_i = u_i$ the i^{th} constraint is an equality. We can also have $l_i = -\infty$ or $u_i = +\infty$. The focus of this paper is on algorithms which can solve problems where some of the constraint functions g_i are nonlinear. Algorithms developed for linear constraints or for problems where f and the g_i must have a special form (e.g. signomials, as in geometric programming) are not included.

Many important algorithms for solving problem NLP have been developed or refined over the last five years. Only the following classes of methods will be discussed (on the assumption that these are the most promising and most suitable for large problems):

(1) Generalized Reduced Gradient (GRG)
(2) Successive Linear Programming (SLP)
(3) Successive Quadratic Programming (SQP).

Descriptions of these methods are brief, but more detailed information is available in [Powell, 1982] and [Powell, 1978].

2. Algorithm Descriptions

2.1 GRG Methods

By introducing appropriate slack variables, the inequality constraints (1) can be converted to equalities. Hence problem NLP can be rewritten as:

minimize f(x)

subject to

$$g(x) = 0 \qquad (3)$$

$$\ell \leq x \leq u \qquad (4)$$

where x now contains both the original and slack variables and g is the vector of all constraint functions.

There are many possible GRG algorithms. Their underlying concepts are described in [Lasdon, Waren, Jain and Ratner, 1978]. This section briefly describes the version currently implemented in GRG2 [Lasdon and Waren,1982], [Lasdon and Waren, 1978].

GRG algorithms use the m equality constraints (3) to solve for m of the variables, called the basic variables (as in L.P.), in terms of the remaining n-m nonbasic variables. Denoting the basic and nonbasic variables as x_1 and x_{nb} respectively, the constraint equations become

$$g(x_1, x_{nb}) = 0. \tag{5}$$

The Jacobian matrix of g may be similarily partitioned as

$$\frac{\partial g}{\partial x} = \left(\frac{\partial g}{\partial x_1}, \frac{\partial g}{\partial x_{nb}} \right) = \left(B_1, B_{nb} \right) \tag{6}$$

where for simplicity the variables are assumed renumbered so that the basics are the first m components of x.

Let $\hat{x}$ be the current feasible point (i.e. $\hat{x}$ satisfies constraints (3) and (4)). Then the specific variables to be chosen as basic must be selected so that B_1, evaluated at $\hat{x}$, is nonsingular. In this case, the constraints (5) can be solved (at least conceptually) for x_1 in terms of x_{nb} to yield the basics as a function of the nonbasics, $x_1(x_{nb})$. This representation is valid for all x_{nb} sufficiently near $\hat{x}_{nb}$. The objective function is then reduced to a function of x_{nb} only,

$$f(x_1(x_{nb}), x_{nb}) = F(x_{nb}) \tag{7}$$

and the original problem is transformed to a simpler reduced problem

$$\text{minimize } F(x_{nb})$$

subject to the bounds on x_{nb}. The function F is called the reduced objective and its gradient F is the reduced gradient.

GRG algorithms solve the original problem (3)-(4) by solving (perhaps only partially) a sequence of reduced problems, which are usually solved by a gradient method. At a given iteration, with nonbasic variables $\hat{x}_{nb}$, the reduced gradient is computed as follows:

(a) Solve

$$B_1{}^t u = \partial f / \partial x_1 \tag{8}$$

for the simplex multiplier vector u ;

(b) Compute the reduced gradient

$$\frac{\partial F..}{\partial x_{nb}} = \frac{\partial f..}{\partial x_{nb}} - u^t B_{nb} \tag{9}$$

Note that all partial derivatives are evaluated at the current point $\hat{x}$.

Following [Murtaugh and Saunders, 1978] the nonbasic variables are further partitioned into s superbasic variables, x_2, which are strictly between their bounds and n-m-s remaining nonbasic variables, x_3, which are at one of their bounds. In GRG2, the reduced gradient with respect to the nonbasic variables, $\partial F/\partial x_3$, is used only to determine if one of these variables should be released from its bound, to join the superbasic set. The reduced gradient with respect to the current superbasics, $\partial F/\partial x_2$, is used to form a search direction, $\hat{d}$. Both conjugate gradient and variable metric methods have been used to derermine $\hat{d}$. Then a one-dimensional search is initiated, whose goal is to solve the problem

$$\underset{a \geq 0}{\text{minimize}}\ F(\hat{x}_{nb} + a\,\hat{d}) \tag{10}$$

In the above $\hat{d}$ has been extended to include zero components for the nonbasic variables at bounds. This minimization is done only approximately, and is accomplished by choosing a sequence of positive values $\{a_1, a_2, \ldots\}$ for a. For each value a_i, $F(\hat{x}_{nb} + a_i\hat{d})$ must be evaluated. By (5), this is equal to

$$f(x_1(\hat{x}_{nb} + a_i\hat{d}),\ \hat{x}_{nb} + a_i\hat{d})$$

so the basic variables $x_1(\hat{x}_{nb} + a_i\hat{d})$ must be determined. These satisfy the system of equations

$$g(x_1,\ \hat{x}_{nb} + a_i\hat{d}) = 0$$

where $\hat{x}_{nb}$, a_i, and $\hat{d}$ are known and x_1 is to be found. If x_1 appears nonlinearly in any constraint then this system must be solved by an iterative procedure such as Newton's method.

An important feature of this algorithm is its attempt to return to the constraint surface at each step in the one dimensional search. This differs from earlier strategies suggested by [Abadie and Carpentier, 1969] and by [Luenberger, 1973], which involve linear searches on the tangent plane to the constraint surface prior to returning to that surface. GRG2 returns to the surface each time because it is simpler and it is felt that it leads to a more reliable algorithm. Computational experience (see [Schittkowski, 1980]) shows that, if properly implemented, this strategy can be developed into an algorithm that is both reliable and efficient.

2.2 SLP Algorithms

Successive Linear Programming methods linearize any nonlinear functions (objective or constraint) about some base point $\hat{x}$. Upper and lower "step bounds" $\hat{s}$ are imposed on Δx to insure that the errors in the linearization are limited. The resulting LP, with variables Δx, is solved and the point $\hat{x} + \Delta x$ is tested. If it is deemed to represent an improved point then it is accepted as the new $\hat{x}$ and the process is repeated. If not, the step bounds are reduced and the LP is resolved. Note that successive points generated by this procedure need not be feasible, even if the initial point is.

However the amount of infeasibility is reduced as the iterations proceed.

SLP methods have the advantage that they can use existing efficient LP codes as "subroutines". The SLP concept was first described by [Griffith and Stewart, 1961] and was later extended by [Buzby, 1974] and [Boddington, 1979]. Recently, [Palacios-Gomez, Lasdon and Enqquist, 1982] have investigated several SLP strategies and have developed a reasonably efficient algorithm.

2.3 SQP Algorithms

Successive Quadratic Programming methods solve a sequence of quadratic programs (QP's). These are NLP problems of special form, with quadratic objective function and linear constraints. As in SLP, the linear constraints are linearizations of the actual constraints about the current point. The quadratic objective function usually used is $x^t Qx + \nabla f^t x$, where Q is an approximation of the matrix of second partial derivatives of the Lagrangian function,

$$L = f - \lambda g - uh.$$

Here g and h are the vectors of equality and inequality constraints respectively, and λ and u are row vectors of Lagrange multipliers.

This approach can be justified in two ways. First, it can be shown that the solution to the QP problem yields the point one would get by applying Newton's method to the necessary (Kuhn-Tucker) conditions for optimality of the original NLP problem [Powell, 1978]. Second, [Biggs, 1975] has shown that the QP solution yields a point near the trajectory of solutions one would obtain if an exterior penalty function method were used to solve the NLP problem.

Usually, the QP solution is not accepted immediately as the new point, rather it is used to specify a search direction and a one dimensional search is then performed to determine the next point. Note that if one just accepts the solution of the QP as defining the new point, then the algorithm behaves like Newton's method; it converges rapidly near an optimal solution of NLP but may not converge from a poor initial point. If Δx is viewed as a search direction, i.e. the new point x^n is defined by

$$x^n = \hat{x} + a\Delta x,$$

then the convergence properties can be improved. Most authors choose a to yield an approximate minimum of some function ℓ along the direction x. Various functions ℓ have been proposed. [Powell, 1978] discusses two choices; the exact penalty function

$$\ell(x,r) = f + r[\textstyle\sum_i |g_i| - \sum_i \min(0,h_i)] \qquad (13)$$

as well as the Lagrangian function L. The philosophy in (13) is to provide a balance between improving the original objective and satisfying the constraints. The function L is suitable because,

(a) on the tangent plane to the active constraints, L has a minimum at the optimal solution to NLP and

(b) the direction x is a descent direction, i.e. L decreases initially in this direction.

The function in (13) also has property (b) if r is large enough. Furthermore it is minimized at the optimal solution to NLP.

3. Alternative Problem Formulations

There are several general formulations for large nonlinear programs. Each emphasizes a different problem feature. One, which we call the "mostly linear" form, introduced by [Griffith and Stewart, 1961], is

$$\text{minimize} \quad g_o(y) + c^t x \tag{14}$$

subject to

$$lc_i \le g_i(y) + a_i^t x \le uc_i \quad i \in NC \tag{15}$$

$$lc_i \le b_i^t y + a_i^t x \le uc_i \quad i \in LC \tag{16}$$

$$\begin{aligned} lx_i &\le x_i \le ux_i, & i &= 1,nL \\ ly_i &\le y_i \le uy_i, & i &= 1,nN \end{aligned} \tag{17}$$

In the above, y is the vector of "nonlinear" variables, which appear in at least one of the problem functions nonlinearly, and x is the vector of linear variables, those which appear linearly everywhere. The nonlinear constraints (15) and possibly the objective (14) each consist of a nonlinear term g_i plus a linear term. There are also linear constraints (16) and simple bounds (17) on both the linear and nonlinear variables. This form is used by [Murtaugh and Saunders, 1980] in their work on the MINOS/Augmented algorithm, as well as in the work on successive linear and successive quadratic programming to be discussed later. This form emphasizes the "mostly linear" aspect found in many large nonlinear problems in which most of the variables appear linearly and most of the constraints are linear. This is true, for example, of many refinery and chemical plant models: [Buzby, 1974],[Baker and Lasdon, 1982].

Another form which appears in [Beale, 1978] and also emphasizes the mostly linear feature is the following:

minimize

$$\sum_{j \in LV} a_{oj} x_j + \sum_{j \in NV} a_{oj}(y) x_j \tag{18}$$

subject to

$$lc_i \le \sum_{j \in LV} a_{ij} x_j + \sum_{j \in NV} a_{ij}(y) x_j \le uc_i, \quad i \in NC \tag{19}$$

$$lc_i \le \sum_{j \in LV} a_{ij} x_j + \sum_{j \in NV} a_{ij} x_j \le uc_i, \quad i \in LC \tag{20}$$

$$\begin{aligned} lx_i &\le x_i \le ux_i, & i &= 1,nL \\ ly_i &\le y_i \le uy_i & i &= 1,nN \end{aligned} \tag{21}$$

Here the variables x and y still play the role of linear and nonlinear variables, but the y's appear in nonlinear functions $a_{ij}(y)$ which multiply some of the x variables - those with indices in the set NV. The x variables with indices in the set LV have constant coefficients a_{ij}. As before, there are also purely linear constraints (20) and simple bounds (21).

Problems often have fewer nonlinear variables, y, when formulated using the Beale form { (18)-(21) } rather than the additive form { (14)-(17) }. In the additive form, the variables x_i with indices in NV must be counted as nonlinear variables. Many nonlinear problems, such as the "pooling" problem (involving the mixing of intermediate products prior to their final blending) and problems where yields are functions of operating conditions and process inputs, contain terms of the form $a_{ij}(y)x_j$. The Beale form is particularily useful in such instances.

Another problem form - called the "state and control" form - useful when optimizing a system described by a large sparse system of nonlinear equations, can be written:

$$\text{minimize} \quad f(x,u) \tag{22}$$

subject to

$$g(x,u) = 0 \tag{23}$$

and

$$u \geq h(x,u) \geq \ell \tag{24}$$

In the above, x is the vector of "state" variables, describing the system in the steady state, and u is the vector of independent, or control, variables. The equations (23) are readily solvable for x given u and the inequalities (24) can include bounds on the state and control variables, as well as more complex restrictions.

Two important classes of systems are well described by this problem form - interconnections of chemical process units and electric power generating systems. [Berna, Locke, and Westerberg, 1980] use this state and control form in their work on chemical process optimization. Many authors have used it for optimal power flow problems - see [Lasdon and Waren, 1980] for a survey.

4. Comparisons and Conclusions

The tabular material following, summarizes the relative merits of SLP, SQP, and GRG algorithms, focusing on their applications to problems with many nonlinear equality constraints.

Algorithm	Relative Advantages	Relative Disadvantages
SLP	1. Easy to implement	1. May converge slowly on problems with non-vertex optima
	2. Widely used in actual practice	2. Will usually violate nonlinear constraints until convergence,
	3. Rapid convergence when optimum is at a vertex	

	4. Can handle very large problems 5. Does not attempt to satisfy equalities at each iteration	often by large amounts
SQP	1. Usually requires far fewest function and gradient evaluations of all three algorithms 2. Does not attempt to satisfy equalities at each iteration	1.Robustness in practice undetermined 2. Will usually violate nonlinear constraints until convergence, often by large amounts 3. Harder to implement than SLP and needs a good QP solver
GRG	1. Probably most robust of all three methods 2. Versatile 3. Can use existing process models	1. Hardest to implement of all 3 methods 2. Needs to satisfy equalities at each step of the algorithm

One feature appears as both an advantage and a disadvantage - whether or not the algorithms can violate nonlinear constraints of the problem by large amounts during the solution process. SLP and SQP usually will generate points yielding large violations. This can cause difficulties, especially in models with log or fractional power expressions. There, negative arguments for these functions may be generated. Such problems have been documented in [Sarma and Reklaitis,1979], using a complex chemical process as an example. There SLP and some exterior penalty type algorithms failed, while a GRG code - described in [Lasdon, Waren, Jain and Ratner,1978] - succeeded, and was quite efficient. On the other hand, algorithms which do not attempt to satisfy the equalities at each cycle can be far faster than any which does, as pointed out in [Berna, Locke and Westerberg,1980]. The fact that such algorithms usually will satisfy any linear constraints at each iteration should ease the difficulties referred to above, but will not eliminate them.

Clearly, all three algorithms have advantages and disadvantages which will dictate their use in certain situations. For large problems, SLP softwarte is used most widely, since it is relatively easyu to implement, given a good LP system. The GRG code GRG2 of [Lasdon and Waren,1982] is widely used for small to medium problems, but GRG software for large problems is available only for special cases, e.g. optimal power floe problems. SQP is a recent approach and general purpose software is not yet available, although some codes are currently "in process".

Acknowledgement

This work was partially funded by ONR contract number N00014-78-C-0222.

References

1. Abadie, J. and J. Carpentier, "Generalization of the Wolfe Reduced Gradient Method to the Case of Nonlinear Constraints," in Optimization, Academic Press, New York, pp. 37-47 (1969).
2. Baker, T.E. and L.S. Lasdon, "Successive Linear Programming at Exxon," working paper, Dept. of General Business, University of Texas, Austin, TX 78712 (1982).
3. Beale, E. M. L., "Nonlinear Programming Using a General Mathematical Programming System," in Design and Implementation of Optimization Software, H. J. Greenberg, ed., Sijthoff and Noordhoff, Holland (1978).
4. Berna, T.J., M.H.Locke and A.W.Westerberg, "A New Approach to Optimization of Chemical Processes," AICHE J., 26 (2), 37 (1980).
5. Biggs, M.C., "Constrained Minimization Using Recursive Quadratic Programming: Some Alternative Subproblem Formulations," in Towards Global Optimization, L.C.W. Dixon and G.P. Szego (eds), North-Holland,Amsterdam (1975).
6. Bodington, C.E., "Nonlinear Programs for Product Blending," Joint National TIMS/ORSA Meeting, New Orleans, April/May (1979).
7. Buzby, B.R., "Techniques and Experience Solving Really Big Nonlinear Programs," in Optimisation Methods, R.W. Cottle and J. Krarup (eds.), English Universities Press, London (1974).
8. Griffith, R.E. and R.A. Stewart, "A Nonlinear Programming Technique for the Optimization of Continuous Processing Systems," Management Science, 7, 379 (1961).
9. Lasdon, L.S. and A.D. Waren, "GRG2 User's Guide," Dept. of General Business,University of Texas, Austin TX 78712 (1982).
10. Lasdon, L.S.,A.D. Waren,A. Jain and M. Ratner, "Design and Testing of a Generalized Reduced Gradient Code for Nonlinear Programming," ACM TOMS, 4, 34 (1978).
11. Luenberger, D., Introduction to Linear and Nonlinear Programming, Addison-Wesley, 158-162 (1973).
12. Murtaugh, B.A. and M.A. Saunders, "Large-scale Linearly Constrained Optimization," Math. Programming, 14, 41-72 (1978).
13. Murtaugh, B.A. and M.A. Saunders, "The Implementation of a Lagrangian Based Algorithm for Sparse Nonlinear Constraints," report SOL 80-1, Systems Optimization Laboratory, Stanford Univ., Stanford, CA (1980).
14. Palacios-Gomez, F.E., L.S. Lasdon and M. Enqquist, "Nonlinear Optimization by Successive Linear Programming," to appear in Management Science.
15. Powell, M.J.D. (ed.), "A Fast Algorithm for Nonlinearly Constrained Optimization Calculations," in Lecture Notes in Mathematics, 63, Springer-Verlag, Berlin (1978).

16. Powell, M.J.D., "Algorithms for Nonlinear Constraints that Use Lagrangian Functions," Math. Programming, 14 (2), 224 (1978).

17. Sarma, P.V.L.N. and G.V. Reklaitis, "Optimization of a Complex Chemical Process Using an Equation Oriented Model," Tenth International Symposium on Math. Programming, Montreal, Aug. (1979).

18. Schittkowski, K., "Nonlinear Programming Codes-Information, Tests, Performance," Lecture Notes in Economics and Mathemativcal Systems, 183, Springer-Verlag, Berlin (1980).

A HIERARCHICAL APPROACH TO PRODUCTION PLANNING IN FLEXIBLE MANUFACTURING SYSTEMS

KATHRYN E. STECKE
Graduate School of Business Administration
The University of Michigan
Ann Arbor, Michigan 48109

ABSTRACT

The flexible manufacturing concept is increasingly being considered in capacity expansion plans as an efficient and viable alternative to conventional manufacturing processes that allows simultaneous machining of small or medium batches of several different part types. As more of these expensive flexible manufacturing systems (FMS) are being built, the importance of good, efficient management increases. Previous research has defined a set of production planning problems, providing a conceptual framework for an FMS manager when setting up an FMS, before production begins, for efficient production. Several approaches have been suggested, each utilizing different amounts of information, for the solution of these problems.

This paper ties many previous results together to provide a hierarchical approach to solving several of these planning problems. First, the problems are solved at an aggregate level, using only some of the available information. These solutions, as well as some theoretical results that are obtained from the solution of the aggregate problems, are used at the detailed level, where all available data are used to allow the solution of realistic planning problems.

1. INTRODUCTION

A flexible manufacturing system (FMS) consists of a set of numerically controlled (NC) machine tools, connected via automated material handling. One or more computers control most real-time functions, such as the cutting operations, part movements, and tool interchanges. Each machine tool has a limited capacity tool magazine that holds all tools required to perform all operations that can be machined at that machine tool. Very quick, automated tool interchange capability means that there is virtually no machine set-up time between operations to consider. However, a part cannot be routed to a particular machine tool unless it is one of a machine type capable of performing that part's next operation and all cutting tools needed to perform that next operation have previously been loaded into the machine tool's tool magazine.

These requirements indicate a natural decomposition of FMS management functions into two distinct, but related, categories: first, the function of setting up the system before production begins; then, the real-time, on-line control of part flow. This paper is concerned with the first function, that of setting up an FMS for future efficient production.

FMS production management can be more complicated than the management of a conventional system because there are more factors and options to consider. Just as in a job shop, the machines are versatile and several part types can be simultaneously machined. However, each individual part can have several routes through the system; parts are not required to be machined in batches. The machining process could and should be as efficient as a transfer line.

Treated in its entirety, the FMS set-up problem is intractable. However, with efficient, flexible production in mind, the following production problems have been defined (Stecke [1982]):

1. Part type selection:
   From a set of part types, each having production requirements, determine a subset for immediate and simultaneous processing.
2. Machine grouping:
   Partition the machines into machine groups such that each machine in a particular group is able to perform the same set of operations.
3. Production ratios:
   Determine the relative ratios at which the part types selected in problem (1) will be produced.
4. Resource allocation:
   Allocate the limited numbers of pallets and fixtures of each fixture type among the selected part types.
5. Loading:
   Allocate the operations and associated tools of the selected set of part types among the machine groups subject to technological and capacity constraints of the FMS.

Unlike the infrequent set-up of a transfer line, the above planning decisions are made often, for example, whenever production requirements are complete for some part type or a rush order arrives.

In this paper, we focus on two of these problems, the machine grouping problem (2) and the loading problem (5), at several levels of detail. Previously, even for non-automated systems, the grouping problem did not exist, since each operation is usually assigned to one and only one machine. The approach to the loading problem is usually to achieve the best possible balance of the work (operations to be performed) among the machines. However, an FMS is an integrated system of quite versatile machine tools. It seemed that new loading and control policies should be devised that utilize the available flexibility to the system's advantage so that production, machine utilization, and overall efficiency could increase.

To this end, various loading objectives and scheduling strategies were developed and applied to data from an existing FMS, with encouraging results. Several objectives performed much better than that of balancing the workload, as measured by the production rate, despite producing highly unbalanced systems (Stecke [1977], Stecke and Solberg [1981b]).

These results spurred subsequent research by the author to investigate FMS grouping and loading problems. The various approaches, models, results, and solutions are tied together in this paper. In the following section, the hierarchy of problems and solutions is given. First, the aggregate grouping and loading problems are defined and previous results are reviewed. Then similar information is given for the detailed grouping and loading problems. The interface between the theoretical results and solutions at the aggregate level and the detailed level models is also given in §2. The use of the aggregate level results and the detailed level models to solve realistic grouping and loading problems is described in §3. Suggestions for future research and further refinements are provided in §4.

2. THE HIERARCHY: AGGREGATE AND DETAILED LEVELS

At the aggregate level, the model used to represent an FMS is a closed queueing network (CQN). The model was chosen for several reasons. First, it models congestion due to different parts waiting for the same machine tool. Second, CQN have been shown to be robust, even for models requiring distributional assumptions and queue disciplines quite different from the actual system distributions and queue disciplines (Giammo [1976], Lipsky and Church [1977], Rose [1976], Solberg [1977], and Suri [1980]). The observed robustness of steady-state results is also surprising since the model requires the aggregation of most information into average processing times and relative visit frequencies for each machine (or machine group). The measure used to evaluate solutions is the expected production rate.

At the detailed level, the grouping and loading problems were formulated as nonlinear mixed integer programs. This model was chosen for several reasons. First, it allows the consideration of all available information such as the actual processing time, tools required, and tool slots required for each operation. Second, it can result in an optimum solution. The measure used to evaluate solutions is the production rate.

At either level, we assume that it is known which part types (and their relative production ratios) shall next be simultaneously machined.

Aggregate Results

The information that is suppressed at the aggregate level is the actual operation times, tools and tool slots required, and machine types required. We assume without loss of generality that there is a single part type and a single machine type. The aggregate grouping and loading problems are now defined.

Aggregate Grouping Problem: Partition the m machines into g machine groups to maximize expected production.

Machines that are in the same machine group are said to be pooled, are identically tooled, and are therefore physically capable of performing the same set of operations during real time control. Pooling, in conjunction with on-line control, implies that a part, whose next operaton requires one of these pooled machines, need only find one machine free. As a result, system performance can only increase, relative to a no-pooling situation.

Aggregate Loading Problem: Allocate a total amount of work among a system of (grouped) machines to maximize expected production.

The workload allocated to each machine group i is proportional to: the relative visit frequency of a part multiplied by the average processing time of an operation by a machine in group i. The expected production, Pr(g,n,S;X), is defined as a function of g, the number of groups; n, the number of parts; $S = [s_1, s_2, \ldots, s_g]$, s_i is the number of machines in group i (S is a partition of machines into groups); and $X = [X_1, \ldots, X_g]$, X_i is the workload assigned to group i (X is an allocation of work among machines) (see Stecke and Solberg [1981b] for additional details.)

The following models were used to solve these aggregate problems. First, symmetric functions (Berge [1963]) and symmetric mathematical programming (Greenberg and Pierskalla [1970]) were used to obtain some theoretical results. Then, CAN-Q (Solberg [1980]), a computer program that uses an efficient algorithm of Buzen [1971] to solve product-form CQN for steady-state results, was used. The aggregate solutions (Stecke and Solberg [1982]) are now given.

Aggregate Grouping Solution: All possible partitions of m machines into g groups are ordered according to expected production. In particular,

1. Fewer groups are better; i.e., pool as much as possible.
2. If constraints define that g groups of m machines are required, then the more unbalanced configuration provides the larger maximum expected production.

The mathematical equivalent of this result is now given. Order the groups according to increasing size, that is, $s_1 \leq s_2 \leq \ldots \leq s_g$. Define $\max\limits_{X} Pr(g,n,(s_1,\ldots,s_g);X)$ to be the maximum expected production from the system, where $X = [X_1,\ldots,X_g]$.

Then for any integer $K > 0$, we have that

1) $\underset{X}{\text{maximum}}\ Pr(g,n,(s_1,\ldots,s_i+K,\ldots,s_g-K);X)$ is strictly less than

$\underset{X}{\text{maximum}}\ Pr(g,n,(s_1,\ldots,s_i,\ldots,s_g);X)$;

2) $Pr(g,n,(s_1,\ldots,s_i+K,\ldots,s_g-K);\vec{1})$ is strictly greater than

$Pr(g,n,(s_1,\ldots,s_g);\vec{1})$;

3) $Pr(g,n,(s_1,\ldots,s_i,\ldots,s_j,\ldots,s_g);X)$ is strictly greater than

$Pr(g,n,(s_1,\ldots,s_i+K,\ldots,s_j',\ldots,s_g);X)$, for $s_i+K=s_{i+1}+K=\ldots=s_j'$.

Aggregate Loading Solution: The solution is a set of optimal allocation ratios, $X_1^*,X_2^*,\ldots,X_g^*$, or theoretically best ratios at which the g groups should be assigned work, which maximizes expected production. In particular,

1. If each group contains the same number of machines, then a balanced workload per machine is optimal.
2. For a system of unequally sized machine groups, unbalancing the assigned workload per machine is optimal. In addition, if a perfect "unbalance" is not possible, then it is better to overload the larger group than otherwise.

Detailed Results

The detailed problems include actual constraints such as tool magazine capacity, integrality of the decision variables, and machine type requirements. The tools and tool slots required for each operation are needed in order to consider available magazine capacity from not duplicating cutting tool assignments as well as tool slot overlap. The detailed problems are now defined.

Detailed Grouping Problem: Partition the m machines into g groups subject to technological and capacity constraints so as to maximize expected production.

Detailed Loading Problem: Allocate the operations and associated tooling of the chosen set of part types among the machine groups subject to the technological and capacity constraints of the FMS and according to some loading objective.

These problems were formulated as nonlinear mixed integer programs (Stecke [1982]). The solutions are now given.

Detailed Grouping Solution: A nonlinear mixed integer program is solved to find the minimum number of machines, g, required to perform all operations of the selected set of part types. Then the Aggregate Grouping Solution is used to immediately determine the partition of the m machines into g groups that maximizes expected production subject to the constraints.

Detailed Loading Solution: The solution is an assignment of operations and associated tools to machine groups. The following six loading objectives of Table 1 are suggested, each applicable to different types of flexible manufacturing systems; the applicability of each depends on the particular manufacturing situation and problem specifics.

TABLE 1

Loading Objectives

1. Balance the assigned machine processing times.
2. Minimize the number of movements from machine to machine, or equivalently, maximize the number of consecutive operations on each machine.
3. Balance the workload per machine for a system of groups of pooled machines of equal sizes.
4. Unbalance the workload per machine for a system of groups of pooled machines of unequal sizes.
5. Fill the tool magazines as densely as possible.
6. Maximize the sum of operation priorities.

Each objective has been mathematically formulated. Situations in which each might be applicable are described in §3.

Since the nonlinearities are products of 0-1 integer variables, the methodology used to solve these problems exactly involved linearization of the nonlinear terms. Several linearizations were examined and one chosen to apply to data from an existing FMS. Several problems were solved in reasonable time using a standard MIP code. The solutions demonstrated the worth of considering tool duplication and overlap to allow additional magazine capacity despite the consequential increase in the size of the linear integer programs (Stecke [1982]).

3. IMPLEMENTATION

Some of the theoretical results at the aggregate level affect the detailed formulations. Other aggregate results are used as some of the input for the detailed models. Since the grouping problem naturally precedes the loading problem in function (operations are assigned to groups of machines), it is solved first in the hierarchical process.

Realistic Grouping Problems

For the part types under consideration for simultaneous machining, the machine type, tools and tool sizes, and processing time required for each operation are known. We now outline the procedure to follow to solve real grouping problems.

Step 1. For each machine type, determine the total processing time requirement and an upper bound on the total number of tool slots required. Dividing by the number of slots in each tool magazine gives an upper bound on the number of machines of each type required to perform all operations. If the upper bounds on the numbers of required machines equal the minimum numbers of required machines, go to Step 2. (For each machine type, the factors that determine whether the upper bounds are minimum include: the number of slots in the remainder of the above quotient, the amount of tool duplication, and even the number of large tools.) If there are any upper bounds that are, or might be, larger than the minimum number of machines, then the grouping formulation of Stecke [1982] can be applied to the machine type in question (i) to find the minimum number of required machines (g_i).

Step 2. The aggregate grouping solution is used to determine the optimal partition of the m_i machines of type i into g_i groups that maximizes expected production.

A side benefit of this procedure is a candidate loading, or a feasible assignment of the operations and tools among the Σg_i machine groups.

Realistic Loading Problems

The aggregate loading solutions, obtained through the use of the CQN model, only relate to the first, third, and fourth objectives of Table 1, the balancing/unbalancing objectives. We discuss these three objectives shortly. We describe how the aggregate results can be used, and a procedure can be defined to provide solutions at the aggregate level, both being input to the detailed formulations dictated by these three objectives. The nonlinear integer problems derived from objectives 2, 5, and 6 are also discussed. First, we discuss various situations in which each loading objective is applicable.

Applicability of each Loading Objective. Part of the aggregate loading solution indicates that it is best to pool as much as possible; in addition, if maximal pooling is impossible, the best partition of m machines into g groups is known. However, if there are many different cutting tools required for the current set of operations, then perhaps no pooling is possible. On the other hand, perhaps there is enough spare capacity in some tool magazines to allow several operation assignments to more than one machine. We call such multiple operation assignments, partial pooling. The fifth and sixth objectives aim to increase flexibility (even if no pooling is possible) by allowing as many alternative routes through the system as possible.

The second objective, minimize movements, is very different from the others. The idea is to keep a part at its current machine for as long as that machine can perform the next consecutive operations, rather than move it for the purpose of better balanced machine workloads. Then, unnecessary travel time (to the next machine) and waiting time (for the potentially busy next machine to become free) are saved. This objective is seen to be applicable when travel time is long relative to processing time and when the material handling system is a bottleneck. In addition, we note that minimizing movements has proven better than balancing, when both objectives were applied using data from an existing FMS. In this case, the transporters did not slow down operation, travel time was short relative to operation time, and often travel could occur while the next machine was completing its current operation. This superiority of minimizing movements occurred despite the resultant highly unbalanced system and the lack of pooling (Stecke [1977]). In general, additional system performance benefits are realized when the minimize movements objective is applied in conjunction with pooling objectives.

Relationship between the Aggregate and Detailed Loading Problems. There is a relationship between the two levels only for the balancing objectives, 1 and 3, and the unbalancing objective, 4. If the aggregate grouping solution is a balanced configuration, then a balanced loading is also optimal. Then the loading problem is solved by using the nonlinear formulation associated with objective one or three. If the aggregate grouping solution is an unbalanced configuration of groups, then the optimal allocation ratios, X_i^*, must be determined before loading objective 4 can be applied. We now describe how to determine the optimal allocation ratios. To simplify notation, we assume without loss of generality that there is only one machine type, since the algorithm is applied identically for each machine type. Similarly, we assume without loss of generality that there are n identical parts in the system. If n has not yet been determined, the algorithm can be applied once, for a specified range of n.

Algorithm to Determine the Optimal Allocation Ratios

1. Order the g machine groups so that $s_1 \leq s_2 \leq \cdots \leq s_g$.
2. A balanced allocation is $X = [s_1, s_2, \ldots, s_g]$. For each i, the "direction" (either less than, greater than, or equal to s_i) of the optimal allocation, X_i^*, is known. Then, if the middle machine (m/2th) is in group i, a good initial starting point in the search for the optimal ratios is, for some $\varepsilon > 0$,
$$[X_1, \ldots, X_g] =$$
$$[s_1 - (1-i)\varepsilon, \ldots, s_{i-1} - \varepsilon,\ s_i,\ s_{i+1} + \varepsilon,\ s_{i+2} + 2\varepsilon, \ldots, s_g + (g-i)\varepsilon]$$
3. Fix g-2 of the X_K. Vary X_i and X_j, $i < j$. The production function, Pr(m,n,S;X), is evaluated along the hyperplane
$$X_i = \sum_{\substack{K=1 \\ K \neq i,j}}^{g} X_K - X_j = K - X_j,$$
for $X_i/s_i \in [a,1]$ and $X_j \in [K-a, K-1]$, for $a < 1$ (i.e., $a = .7$). The production function can be evaluated by using a computer program that is a variation of Solberg's CAN-Q [1977]. (A listing of the adapted program can be found in Stecke and Solberg [1981a], where several examples are given that demonstrate the use of the program.)
4. The maximum expected production from the optimal allocation ratios has been found. The program provides sensitivity information on the change in the expected production resulting in a change in a workload parameter. This indicates a direction and magnitude of a change in some X_i. Fix a different M-2 of the X_K and return to Step 3. Search along different hyperplanes until the sensitivity analysis indicates that the optimum allocation ratios have been found.

Aggregate Results as Input for the Detailed Models. When the optimal allocation ratios, X_i^*, are found, they are used in the nonlinear objective function formulated from the fifth loading objective, unbalancing. Each loading objective of Table 1 gives rise to a different nonlinear integer program. The formulations, given in Stecke [1982], can be solved using an available integer code, which yields the optimal allocation of operations and tools among machine groups.

4. FUTURE RESEARCH

The solution methodologies reported here have all been optimum-producing. The nonlinear integer problems have been solved for real grouping and loading problems on a large computer system (CDC 6600) using a standard integer code. However, real-time FMS control requires very quick solutions to these planning problems. Future research plans include the development of efficient heuristic algorithms that provide good solutions to FMS grouping and loading problems. These procedures should be executable on minicomputers that are now available in essentially real-time.[1]

---

[1] This research was supported in part by the National Science Foundation Grant No. APR74 15256 and by a grant from the Ford Motor Company.

REFERENCES

1. BERGE, CLAUDE, Topological Spaces, The Macmillian Company, New York, N.Y. (1963).

2. BUZEN, JEFFREY P., "Queueing Network Models of Multiprogramming," Ph.D. Thesis, Harvard University, Cambridge, MA (1971).

3. GIAMMO, T., "Validation of a Computer Performance Model of the Exponential Queueing Network Family," in Proceedings of the International Symposium on Computer Performance Modeling, Measurement, and Evaluation, Harvard University (1976). Also published in Acta Informatica, Vol. 17, No. 2, pp. 137-152 (1976).

4. GREENBERG, HARVEY J. and PIERSKALLA, WILLIAM P., "Symmetric Mathematical Programs," Management Science, Vol. 16, No. 5, pp. 309-312 (January 1970).

5. LIPSKY L. and CHURCH, J. D., "Applications of a Queueing Network Model for a Computer System," Computing Survey, Vol. 9, pp. 205-221 (1977).

6. ROSE, C. A., "Validation of a Queueing Model with Classes of Customers," in Proceedings of the International Symposium on Computer Performance Modeling, Measurement, and Evaluation, Harvard University, Cambridge, MA, pp. 318-325 (1976).

7. SOLBERG, JAMES J., "A Mathematical Model of Computerized Manufacturing Systems," in Proceedings of the 4th International Conference on Production Research, Tokyo, Japan (August 1977).

8. SOLBERG, JAMES J., "CAN-Q Use's guide," Report No. 9 (Revised) NSF GRANT No. APR74 15256, School of Industrial Engineering, Purdue University, W. Lafayette IN 47907 (July 1980).

9. STECKE, KATHRYN E., "Experimental Investigation of a Computerized Manufacturing System," Master's Thesis, Purdue University, W. Lafayette IN 47907 (December 1977).

10. STECKE, KATHRYN E., "Formulation and Solution of Nonlinear Integer Production Planning Problems for Flexible Manufacturing Systems," Management Science, forthcoming (1982).

11. STECKE, KATHRYN E. and SOLBERG, JAMES J., "The CMS Loading Problem," Report No. 20, NSF Grant No. APR 74 15256, School of Industrial Engineering, Purdue University, W. Lafayette IN 47907 (February 1981a).

12. STECKE, KATHRYN E. and SOLBERG, JAMES J., "Loading and Control Policies for a Flexible Manufacturing System," International Journal of Production Research, Vol. 19, No. 5, pp. 481-490 (September-October, 1981b).

13. STECKE, KATHRYN E. and SOLBERG, JAMES J., "The Optimality of Unbalanced Workloads and Machine Group Sizes for Flexible Manufacturing Systems," Working Paper No. 290, Graduate School of Business Administration, The University of Michigan, Ann Arbor MI 48109 (January 1982).

14. SURI, RAJAN, "Robustness of Analytical Formulae for Performance Prediction in Certain Nonclassical Queueing Networks," Technical Report No. 674, Division of Applied Sciences, Harvard University, Cambridge MA 02138 (August 1980).

OPTIMIZATION OF FLEXIBLE MANUFACTURING SYSTEMS USING NEW TECHNIQUES IN DISCRETE EVENT SYSTEMS*

RAJAN SURI and XIREN CAO
Harvard University, Division of Applied Sciences, Cambridge, MA

ABSTRACT

Two new methods, the phantom customer method and the marked customer method, are introduced as extensions of the perturbation analysis approach to discrete event systems. These methods enable efficient and accurate optimization of multiclass queueing networks with general service-time distributions. A useful practical application of these methods is for optimization of Flexible Manufacturing Systems performance.

1. MOTIVATION AND BACKGROUND

Flexible Manufacturing Systems (FMS) are proving to be useful for improving the productivity of discrete parts manufacturing [1,2]. However, these systems are also very complex and thus difficult to operate efficiently [3]. Here we shall be concerned with optimizing various FMS design/operating parameters. (We assume familiarity with basic FMS terminology.) Typical decision parameters which affect FMS performance are: a) machining times, b) allocation of work to machines, c) number of pallets/fixtures for each part type, d) buffer storage sizes, and e) machine failure- and repair-time distributions.

Under certain assumptions, an FMS can be modelled as a closed queueing network, and recently several researchers have shown that analytical queueing models are useful in the planning and control of FMS operations [4-9]. These analytical models are however, based on unrealistic assumptions for FMS. Although some theoretical justification can be given for their use in FMS modelling, [10], and some heuristic extensions can be used to improve their performance [11], it is still true that these models become inaccurate when certain factors are significant. For example, they could not be easily used to study the effect of buffer sizes or repair-times.

A more accurate alternative to these analytic models is to use a detailed simulation to predict FMS performance. Normally, in optimizing some Performance Measure (PM), one changes each parameter and computes the PM sensitivity by doing a new simulation after each change. This gives an estimate of the gradient vector of the PM. The gradient can then be used along with any mathematical programming algorithm to get a new set of parameters, and in this manner, to iteratively optimize the PM. The disadvantage of this method is that, for N parameters, each iteration step requires N new simulations. Now, a simulation can take (typically) over 100 times the computer time that a queueing model would use. This method can therefore be computationally very demanding.

Recently, an efficient alternative to this 'brute-force' simulation

---

*This work was supported by the U.S. Office of Naval Research Contracts N00014-75-C-0648 and N00014-79-C-0776, and by the National Science Foundation Grant ENG78-15231. The authors are grateful to Prof. Y.C.Ho of Harvard University for the opportunity to conduct this research.

method has been developed, namely the perturbation analysis approach of Y.C.Ho and colleagues (references in next section). This approach retains the precision of a detailed simulation model, while incorporating some efficiency by using analysis. It has been successfully applied to the performance optimization of serial production lines. In networks, such as FMS, some new issues arise. This paper addresses some of these issues, and in particular extends the perturbation analysis approach to multi-class queueing networks with blocking and general service times, of which an FMS is a practical example.

2. BRIEF REVIEW OF PERTURBATION ANALYSIS

The perturbation analysis approach to discrete event systems optimization was introduced by Ho et.al. [12], in the context of serial production lines (transfer lines). It was formalized and elaborated for such applications in [13-15]. The first generalization of this method to networks of queues was given in [16]. This paper extends [16] to cover certain parameters that are particularly important for FMS.

2.1 Basic Idea

The basic idea of the Perturbation Analysis Method (PAM) is to consider a given sample path of a discrete event system, which is called the nominal sample path. (This sample path could be obtained from a detailed simulation, or from observations/experiments on the actual system.) Now consider the following question: if the time of occurrence of a specific event in the nominal path had been perturbed, what would have been the effect on the (various) PMs of the system at the end of this observation period, for this perturbed path? PAM allows us to answer this question efficiently, for a large set of possible perturbations: it gives us a set of simple rules, which specify how a given perturbation would have affected other events during the observation period; these are called perturbation propagation rules.

Next, let us see how this is useful for parameter optimization. Consider a decision parameter, say R = mean repair time of machine M, and suppose we are interested in the sensitivity of the system production rate to changes in R. Then, we are interested in answering the question: if R had been R+dR, what would have been the corresponding production rate for the perturbed path? In the nominal path, each repair time of M, say r[i], would have been perturbed by dr[i]. The value of dr[i] depends on dR and on the repair-time model being used [17], and is called the perturbation generation rule. Next, each perturbation that is generated, is propagated along the nominal path according to the propagation rules, and the various perturbations are added together using superposition rules, to determine the net effect on the PM [13-16]. Finally the results can be appropriately interpreted in a stochastic context, to make statements which hold for more than just the sample path observed [18].

All the above calculations can be implemented on a single simulation run (or experiment), for little added computational effort. Thus, effectively, we can compute a whole gradient vector in almost the same time as one simulation, which is an N-fold increase in efficiency over the 'brute-force' method of §1.

2.2 Extension to FMS Optimization

In [13-16] it is shown how PAM can be used to derive gradients w.r.t. (a) service times (for any distribution), and (b) buffer sizes. In an FMS, another important decision is the number of pallets/fixtures allocated to the production of a given part-type [6]. Optimization of this decision requires us to compute gradients w.r.t. the number of jobs of a given type that are circulating in the FMS. (Traditionally, this would be called the number of customers of a given class in a multiclass network.) Previous work on PAM does not cover this aspect, which will be studied here. We shall see that this involves some non-trivial extensions to existing PAM theory: we introduce a technique called the phantom customer method (and its dual, the marked customer method), which require re-stating some of the perturbation generation and propagation rules of PAM. When combined with existing PAM theory, our method gives an efficient way of optimizing FMS performance, particularly w.r.t. decision parameters that are not adequately covered by the queueing models.

3. THE PHANTOM CUSTOMER AND MARKED CUSTOMER METHODS: AN OVERVIEW

For generality, we shall use the terminology of queueing networks, rather than of FMS, in this section. Also, we restrict ourselves here to study of throughput (although PAM can be used for other performance measures too [15]). Thus we are interested in the gradient of throughput (of each class) with respect to a change in the number of jobs (of each class).

3.1 The Marked Customer Method

In this method, for each class, we mark one customer of that class, in the simulation. The aim is to predict the system performance if this customer were not present in the system. Thus, as the marked customers move around the network in the simulation, we generate perturbations according to rules to be elaborated below. For example, each time a marked customer is served, the customer behind (if any) would have been served earlier if the marked customer had not been present. So this customer's start-of-service time would be perturbed. Each of these perturbations is then propagated in the manner that is usual for PAM. However, two points need to be noted. Firstly, in this approach, the presence of marked customers requires certain re-formulation of the existing perturbation generation and propagation rules. Secondly, the perturbations generated by this approach, since they correspond to removal of entire service times, are rather large compared to those in previous PAM research. Therefore, in order to get accurate results it is necessary to use more complex, higher order propagation rules [19]. These two points distinguish the approach in this paper from previous PAM work, and will be detailed below.

3.2 The Phantom Customer Method

This can be thought of as the 'dual' of the marked customer method, since it estimates the gradient by predicting the effect of an increase in the number of customers of any given class. For each class, this method puts one additional phantom customer into the network. The phantom circulates according to the routing rules of its class, but its service time is always zero at every server. It is, however, used to generate appropriate perturbations. For example, when a phantom arrives at a server, if the phantom had been a 'real' customer, then the server would have had to provide some service to it, before serving the next customer. This means the next customer's start-of-service time would have been perturbed by some amount. Perturbations so generated (the rules are actually more complex) are then propagated analogous to the marked customer method. Note that the reason for using phantoms is that they enable us to predict (without influencing the nominal path) the rate at which the extra customer would have demanded service.

The next section gives in detail the theory behind our method. (Readers interested mainly in applications/results could turn directly to §5.)

4. THEORY OF THE METHOD

We now derive the detailed perturbation generation and propagation rules for our method. Here we give only the rules for the marked customer method, since the phantom customer method is analogous. In addition, since effects of service times and buffer sizes have been analyzed previously [12-16], we restrict our analysis to changes in customer population.

Since PAM has been explained in detail elsewhere [12-19] we assume familiarity with its concepts. Consider a closed queueing network in which M customers are circulating between K servers. Suppose during the experiment time T, a total of Nsvd[i] customers are served by server i, and among them Nmc[i] are marked customers. Then the throughput of server i in the Nominal path is

$$TP_N[i] = \frac{Nsvd[i]}{T} \tag{1}$$

Now suppose we could predict Pert[i] = the total perturbation at server i at the end of the experiment. (The perturbation at a server is the difference in occurrence times of a specified event, for the nominal versus perturbed paths, see [12-19].) Then the predicted throughput of server i in the Perturbed path (with M-1 circulating customers) would be

$$TP_P[i] = \frac{Nsvd[i] - Nmc[i]}{T - Pert[i]} \tag{2}$$

Calculation of Pert[i] by PAM requires two steps of analysis, as mentioned above, namely perturbation generation and perturbation propagation. The details of these two steps now follow. (We will assume there is only one marked customer, since the same analysis can be done in parallel for several marked customers of different classes.)

4.1 Perturbation Generation Rules

(i) Whenever the marked customer (MC) is served by a server, and is not the only customer there, a perturbation of Smc (=current observed Service time of MC) is generated at this server. The reason is simply that, if the MC were not present, the next customer would have received its service earlier by an amount Smc (Fig.1).

(ii) If the MC is the only customer at some server, then a new No Input (NI) would be created if the MC were not there. There are three subcases for this:

(ii.a) In the situation in Fig.2a, if the MC were not present a new NI would appear at server i, and the next customer would receive its service at time t1 instead of t0. So the perturbation is no longer Smc. We call this as a 'Partial Perturbation', PP = t0-t1.

(ii.b) In the case in Fig.2b, eliminating the MC only extends the NI of server i, no perturbation is created. So

$$NI(P) = NI(N) + Smc \tag{3}$$

where NI(P) and NI(N) are the NI durations in the Perturbed and Nominal paths respectively.

(ii.c) This case in Fig.2c combines cases (ii.a) and (ii.b). When in the nominal path the MC terminates a NI of some server, this server would get both a partial gain and an extended NI duration in the perturbed path, so

$$PP = t0 - t1 \quad ; \quad NI(P) = NI(N) + Smc - PP \tag{4}$$

4.2 Perturbation Propagation Rules

The basic propagation rules used in the MC method are the first order propagation rules specified in [19]. However, we also need to consider some special cases, e.g., in some cases the perturbations cause first order propagation in the perturbed path but second order propagation in the nominal path. Specifically, the perturbation propagation rules in the marked customer method are:

(i) When an NI is not terminated by the MC, the ordinary first order propagation rules are applied.

(ii) The effect of the MC is considered in three respects:

(a) The elimination of the MC will create a new NI or extend the duration of the old NI, so this has to be considered for the first order propagation rules. (cf. potential NI in first order rules).

(b) If the MC terminates a NI, then no perturbation will be propagated, since this customer would not have been present.

(c) For customers immediately behind the MC at a server, we propagate perturbations by first order rules, except we assume this MC was not present in the system. e.g., in Fig.3 customers from server k or server j may eliminate the new NI the MC created.

5. EXPERIMENTAL RESULTS

Use of the MC method, and its prediction accuracy, are illustrated by some examples. The first set of examples considers 'classical' networks (with exponential service times and no blocking), for which analytical formulae exist for the throughput.

5.1 Verification Against Classical Networks

This example considers a network with three servers, each having an exponentially distributed service time with a mean of 10 units. Each simulation run consisted of 100,000 services. Details of routing probabilities and simulation observations are in [20]. The results for two different cases, with 4 customers and with 9 customers, are in Table 5.1.

5.2 Validation Against General Networks (such as FMS)

Now we consider examples with general service time distributions. For such networks, no analytical formulae are available, and we must resort to simulation to verify the predictions. The network consists of three servers, whose service times are uniformly distributed over the range [9.5, 10.5]. This could correspond to an FMS with a slight variation in machining times. Again, details of routing and simulation output can be found in [20]. Results of three sets of predictions are shown in Table 5.2.

6. CONCLUDING REMARKS

From Tables 5.1-5.2 we see that the MC method has an accuracy of around 1% for predicting the new throughput value, and an accuracy of 5% for predicting the sensitivity. These values are similar to the prediction accuracy that would be obtained by doing another simulation at the new value (since each simulation would also suffer from statistical variation). Hence preliminary tests show that the accuracy of the MC method is well within acceptable limits, and furthermore, it has the advantage of being computationally efficient. More comprehensive tests are currently being conducted and will be available in [20].

REFERENCES

[1] Cook, N.H., 'Computer-managed Parts Manufacture', Scientific American 232, No.2, 1975.

[2] Hughes, J.J., Hutchinson, G.K. and Gross, K.E., 'Flexible Manufacturing Systems for Improved Mid-Volume Productivity', Proc. 3rd AIIE Systems Engineering Conference, 1975.

[3] Hutchinson, G.K., 'The Control of Flexible Manufacturing Systems: Required Information and Algorithm Structures', IFAC Symposium on Information-Control Problems in Manufacturing Technology, Tokyo, 1977.

[4] Solberg, J.J., 'A Mathematical Model of Computerized Manufacturing Systems', Proc. 4th Intl. Conf. Prod. Res., Tokyo, Japan, 1977.

[5] Stecke, K.E., Production Planning Problems for Flexible Manufacturing Systems, Ph.D. Dissertation, Purdue University, W. Lafayette, IN, August 1981.

[6] Hildebrant, R.R., 'Scheduling Flexible Machining Systems Using Mean Value Analysis', Proc. IEEE Conf. Decision and Control, Albuquerque, NM, 1980.

[7] Suri, R., 'New Techniques for Modelling and Control of Flexible Automated Manufacturing Systems', Proc. IFAC 8th Triennial World Congress, Vol. 14, pp.175-181, Kyoto, Japan, 1981.

[8] Kimemia, J.G. and Gershwin, S.B., 'Multicommodity Network Flow Optimization in Flexible Manufacturing Systems', Report ESL-FR-834-2, LIDS, M.I.T., Cambridge, MA, April 1980.

[9] Buzacott, J.A. and Yao, D.D.W., 'Flexible Manufacturing Systems: A Review of Models', Working Paper 82-007, Dept. of Indust. Engg., Univ. of Toronto, March 1982.

[10] Suri, R., 'Robustness of Queueing Network Formulae', J. ACM, to appear in July, 1983.

[11] J.-B. Cavaille and D. Dubois, 'Heuristic Methods Based on Mean Value Analysis for Flexible Manufacturing Systems Performance Evaluation', Proc. 21st IEEE Conf. Decision and Control, Orlando (FL), Dec. 1982 (to appear).

[12] Y.C. Ho, M.A. Eyler, and T.T. Chien, 'A Gradient Technique for General Buffer Storage Design in a Serial Production Line', Int. J. Prod. Res. 17, 6, 1979, 557-580.

[13] Y.C. Ho, M.A. Eyler, and T.T. Chien, 'A New Approach to Determine Parameter Sensitivities on Transfer Lines', Mgmt. Sci., 1983 (to appear).

[14] Y.C. Ho and C. Cassandras, 'A New Approach to the Analysis of Discrete Event Dynamic Systems', Automatica, 1982 (to appear).

[15] Cassandras, C., Sample Path Analysis of Discrete Event Dynamic Systems, Ph.D. Thesis, Division of Applied Sciences, Harvard University, Cambridge, MA, October 1982.

[16] Y.C. Ho and X. Cao, 'Perturbation Analysis and Optimization of Queueing Networks', J. Optim. Theory Applic., to appear in Vol.40, No.4, August 1983.

[17] Suri, R., 'Implementation of Sensitivity Calculations on a Monte-Carlo Experiment', J. Optim. Theory Applic., to appear in Vol.40, No.4, August 1983.

[18] Ho, Y.C., Cassandras, C. and Suri, R., 'Stochastic Similarity and Linearity in Perturbation Analysis of Discrete Event Dynamic Systems', Division of Applied Sciences, Harvard University, Technical Report, to appear.

[19] Ho, Y.C., Cao, X. and Cassandras, C., 'Infinitesimal and Finite Perturbation Analysis in Queueing Networks', Proc. 21st IEEE Conf. Decision and Control, Orlando, FL, December 1982.

[20] Suri, R. and Cao, X., 'The Phantom Customer and Marked Customer Methods for Optimization of Multiclass Queueing Networks with Blocking and General Service Times', Division of Applied Sciences, Harvard University, Technical Report, to appear.

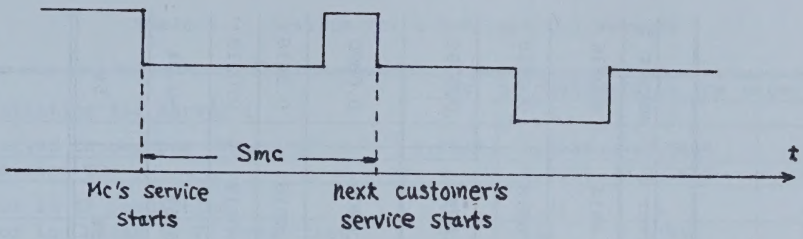

Smc
t
Mc's service starts
next customer's service starts

Fig. 1

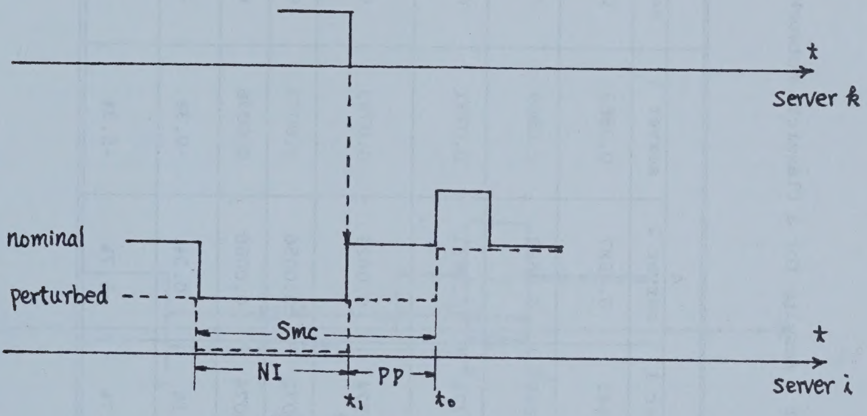

t
Server k
nominal
perturbed
Smc
NI
PP
t₁
t₀
t
server i

Fig. 2a

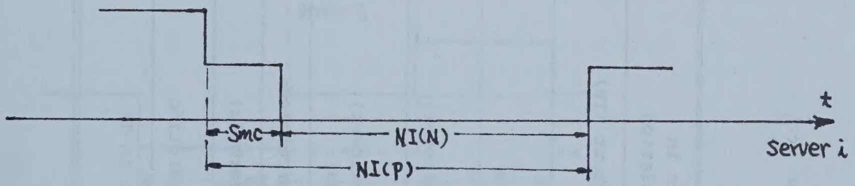

t
Smc
NI(N)
NI(P)
server i

Fig. 2b

Table 5.1 Results for a Classical Network

No. of customers in nominal simulation	4			9		
	server 1	server 2	server 3	server 1	server 2	server 3
Observed Throughput (TP) (nominal)	0.0845	0.0687	0.0863	0.0815	0.0814	0.0816
Exact TP (from formula)	0.0848	0.0689	0.0869	0.0818	0.0818	0.0818
Predicted TP (one less customer)	0.0773	0.0631	0.0791	0.0796	0.0795	0.0798
Exact TP (one less customer) (from formula)	0.0774	0.0629	0.0793	0.0800	0.0800	0.0800
Change in TP (predicted)	0.0072	0.0056	0.0072	0.0018	0.0018	0.0018
Change in TP (formula)	0.0074	0.0060	0.0076	0.0019	0.0019	0.0019
Error in TP Prediction	-0.1%	0.3%	-0.3%	-0.5%	-0.6%	-0.3%
Error in Change in TP predictions	-2.7%	-6.7%	-5.3%	5%	5%	5%

Table 5.2 Results for a Nonclassical Network

Statistics for Server 1	No. of customers in the network			
	4	3	2	1
Observed throughput (TP)	0.0998	0.0980	0.0666	0.0333
predicted TP	-	0.0978	0.0660	0.0334
Error in TP prediction	-	0.2%	1%	0.3%
Error in Change in TP prediction	-	11%	1.9%	0.3%

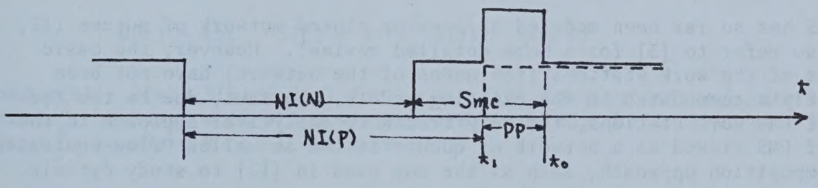

NI(N)
Smc
NI(P)
PP
t
t_1
t_0

Fig . 2c

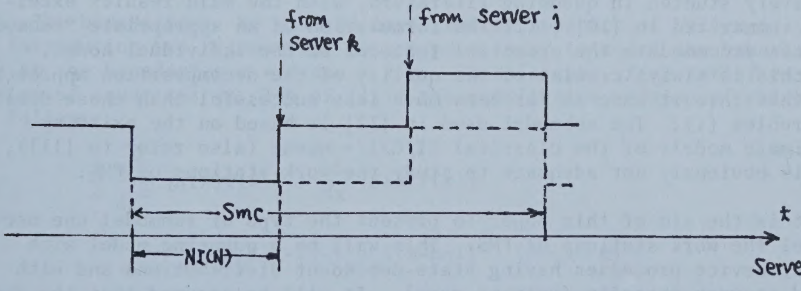

from Server k
from Server j
Smc
NI(N)
t
Server i

Fig. 3

MODELING A MULTI-MACHINE WORK STATION IN FLEXIBLE MANUFACTURING SYSTEMS (FMS)

DAVID D.W. YAO & J.A. BUZACOTT
Dept. of Industrial Engineering
University of Toronto
Toronto, Ontario
Canada M5S 1A4

ABSTRACT

A diffusion approximation model is developed to accommodate the basic features of the work stations in FMS. The model is then used to study some important problems at the work station level.

1. INTRODUCTION

In FMS planning a recognized fact is that the pooling of parallel machines (organized as a work station) may yield improved system performance ([14]). Basic features of such a multi-machine work station are: (1) state-dependent input (due to computer control) and output (due to the multiple machines) processes; (2) limited in-process storage capacity; (3) non-Markovian interarrival/service time distributions (in general).

FMS has so far been modeled as open or closed network of queues ([2, 13]; also refer to [3] for a more detailed review). However, the basic features of the work stations (the nodes of the network) have not been adequately accommodated in the existing models. In fact, due to the features of the work stations, the only tractable analytical approach to the study of FMS viewed as a network of queues is the so called "flow-equivalent" or decomposition approach, such as the one used in [12] to study dynamic job shops viewed as open network of queues.

A decomposition approach involves two main problems: (1) The formulation of the first two moments of the equivalent flows. This has been extensively studied in queueing literature, with the main results excellently summarized in [10]. (2) The formulation of an appropriate "submodel" which can accommodate the essential features of the individual nodes. While this is always crucial to the quality of the decomposition approach, researches into it have so far been much less successful than those dealing with problem (1). The submodel used in [12] is based on the existing approximate models of the classical $GI/G/1/\infty$ queue (also refer to [11]), which is obviously not adequate to study the work stations of FMS.

It is the aim of this paper to present the type of submodel one needs to model the work stations of FMS. This will be a queueing model with arrival/service processes having state-dependent distributions and with limited storage capacity (waiting room). It will be assumed that the first two moments of the interarrival/service time distributions are given. To be specific, we will assume that these distributions will be fully characterized by their rates: $\lambda(k)$, $\mu(k)$ and squared coefficients of variation (scv): $A(k)$, $S(k)$; where, $k = 0,1,\ldots,m$ is the number of jobs in the station (including those in service), with m being the maximum allowed. This assumption is not only based upon considerations of practical interest but also upon the fact that two-parameter approximations of point processes are well established techniques in literature ([15]). Our modeling approach is the diffusion approximation. For brevity we omit the preliminaries of this approach. They can, however, be found in texts such as [1, 5, 7].

2. THE MODEL AND SOLUTIONS

We consider the steady-state stochastic process of the number of customers in the station: $\{Q | Q = 0,1,\ldots,m\}$, and denote its probability densities by $\pi_k \equiv \Pr[Q = k]$, $(k = 0,1,\ldots,m)$. The essence of our approach is to approximate the discrete queueing process $\{Q\}$ by a continuous diffusion process $\{X | X \in [0,m]\}$, the pdf of which we shall denote by $f(x)$. We further characterize $\{X\}$ as follows.

(1) The drift $\beta(x)$ and the variance $\alpha(x)$: These are of course state dependent. In order to facilitate the solution to the diffusion equation, they are formulated as piecewise continuous functions (this approach has also been used in [8] for an M/G/s/∞ model). For $0 < x < 1$ and $k-1 \le x < k$, where $k = 2,\ldots,m$, we have

$$\beta(x) = \lambda(k-1) - \mu(k) \equiv \beta_k \tag{2.1}$$

$$\alpha(x) = \lambda(k-1)A(k-1) + \mu(k)S(k) \equiv \alpha_k \tag{2.2}$$

where (2.2) is formulated by assuming the input/output processes of the station are renewal and then using the asymptotic results of renewal theory ([4]). Consider two special cases: (i) s parallel servers (machines) each with fixed rate μ and scv c_s^2, we have

$$\mu(k) = k\mu, \quad S(k) = 1+(c_s^2-1)/k \quad (k \le s) \tag{2.3}$$

and $\mu(k) = \mu(s)$, $S(k) = S(s)$ for $k > s$. (ii) Constant arrival rate λ and scv c_a^2; however, when all s servers are occupied, the arrival rate is tuned down to λq, where $0 < q < 1$, we have

$$\lambda(k) = \lambda q, \quad A(k) = qc_a^2+(1-q) \quad (k \ge s) \tag{2.4}$$

The derivations are quite straightforward and can be found in [15].

(2) The boundaries: We use the more general approach of Feller ([1, 6]) by introducing two "elementary return processes" to model the behavior of (X) at the boundaries $x = 0$ and $x = m$. In this case, $f(x)$ satisfies a system of equations (p. 146 of [1]), the equilibrium version of which is as follows:

$$\frac{1}{2}\frac{d^2}{dx^2}[\alpha(x)f(x)] - \frac{d}{dx}[\beta(x)f(x)]$$

$$= -\pi_0 h_0 \delta(x-1) - \pi_m h_m \delta(x-m+1) \quad (0 < x < m) \tag{2.5}$$

$$\lim_{x\to 0}\left[\frac{1}{2}\frac{d}{dx}[\alpha(x)f(x)] - \beta(x)f(x)\right] = \pi_0 h_0 \tag{2.6}$$

$$\lim_{x\to m}\left[\frac{1}{2}\frac{d}{dx}[\alpha(x)f(x)] - \beta(x)f(x)\right] = -\pi_m h_m \tag{2.7}$$

$$\lim_{x\to 0} f(x) = \lim_{x\to m} f(x) = 0 \tag{2.8}$$

where $\delta(x-1)$ and $\delta(x-m+1)$ are Dirac's delta functions; h_0^{-1} and h_m^{-1} are mean holding times at the boundaries 0 and m, respectively. It should be noted that the holding times at the boundaries are generally non-Markovian. However, it has been shown in [7] that in equilibrium, only the first moment of the holding-time distribution appears in the diffusion equations. Since it is generally impossible to give exact derivations of h_0 and h_m, we give the following heuristic formulations. If $\{X\}$ hits the boundary $x = m$ $(x = 0)$ frequently, as it does in the case of heavy (light) traffic, the mean holding time can be approximated by the mean interdeparture (interarrival) time: $h_m = \mu(m)$ $(h_0 = \lambda(0))$; if otherwise, the mean holding time is better approximated by the mean residual interdeparture (interarrival) time: $h_m = 2\mu(m)/[S(m)+1]$ $(h_0 = 2\lambda(0)/[A(0)+1])$.

Assume $\beta_k \neq 0$, and let $\gamma_k \equiv \alpha_k/(2\beta_k)$, $(k = 1,\dots,m)$. The solution to the diffusion equations can be derived as follows:

$$f(x) = \begin{cases} (\pi_0 h_0/\beta_1)[\exp(x/\gamma_1)-1] & (0 < x < 1) \\ \pi_0\theta_k\exp((x-k)/\gamma_k) & (k-1 \le x < k,\ k = 2,\dots,m-1) \\ (\pi_m h_m/\beta_m)[1-\exp((x-m)/\gamma_m)] & (m-1 \le x < m) \end{cases} \tag{2.9}$$

From the continuity of $f(x)$, we have

$$\begin{aligned} \theta_1 &= (h_0/\beta_1)[\exp(\gamma_1^{-1})-1] \\ \theta_{k+1} &= \theta_k\exp(\gamma_{k+1}^{-1}) \quad (k = 1,\dots,m-1) \\ \pi_m &= \pi_0\beta_m\theta_m/[h_m(\exp(\gamma_m^{-1})-1)] \end{aligned} \tag{2.10}$$

From the normalization condition $\pi_0 + \int_0^m f(x)dx + \pi_m = 1$, π_0 can be derived as follows:

$$\pi_0 = [1-h_0/\beta_1 + \theta_m(1+\beta_m/h_m)/(\exp(\gamma_m^{-1})-1) + \sum_{k=1}^{m-1}\theta_k(\gamma_k-\gamma_{k+1})]^{-1} \tag{2.11}$$

The mean number of jobs in the station (including those in service) can be approximated by

$$\begin{aligned} L_s &= \int_0^m xf(x)dx + m\pi_m \\ &= \pi_0[(h_0/\beta_1)(\gamma_1 - \tfrac{1}{2}) + \sum_{k=1}^{m-1}\theta_k(\gamma_k-\gamma_{k+1})(k-\gamma_k-\gamma_{k+1})] + \\ &\quad + \pi_m[(h_m/\beta_m)(m-\gamma_m - \tfrac{1}{2})+m] \end{aligned} \tag{2.12}$$

If $\beta_k = 0$ for some k, because of the continuity of $f(x)$, (2.11) and (2.12) are still valid by letting $\beta_k \to 0$ and taking the limit. In the case of a work station having s parallel machines, the probability that at least one machine is not occupied (P_{id}) is often crucial to the derivation of other performance measures. We give the following approximation:

$$P_{id} = \pi_0 + \ldots + \pi_{s-1} = \pi_0 + \int_0^{s-1+d} f(x)dx$$

$$= \pi_0[1-h_0/\beta_1 + \sum_{k=1}^{s-1} \theta_k(\gamma_k-\gamma_{k+1})+\theta_{s-1}\gamma_s \exp(d/\gamma_s)] \qquad (2.13)$$

where d = 0, 1/3 and 1/2 for heavy, moderate and light traffic, respectively. This "compensation" of probability densities is mainly based on the fact that f(x) is forced to vanish at the origin (cf. (2.8)), yielding quite small values for π_k of small k's, which affects the accuracy unless the traffic is heavy. Once P_{id} is derived, other performance measures follow directly, for instance, the mean number of busy servers:

$$BS = \int_0^{s-1+d} xf(x)dx + s(1-P_{id}) \qquad (2.14)$$

the throughput: TH = μBS; the mean waiting time in queue (not including service time): $W_q = (L_s-BS)/TH$, etc.

3. NUMERICAL EXAMPLES

In Tables 1 and 2 we compare numerical results of our model with those of regenerative simulation (with a 95% confidence interval) for two types of stations with various parameters. The implementation of the model in the context of network of queues is currently under way and will be reported in another paper. Here we focus on studying the work station submodel and demonstrate through two examples the kind of insights we can get at this level.

Example 1. Consider an $M/E_k/s/m$ type work station. Formula (8) of [9] gives the following approximation for the mean waiting time in the M/G/s/m queue (in our notation):

$$W_q(NR) = [\tfrac{1}{2}(c_s^2+1)(1-\pi_m-P_{id})-(m-s)\pi_m]/[(1-\pi_m)(s\mu-\lambda)] \qquad (3.1)$$

And the replacement of π_m and P_{id} of (3.1) by the corresponding probabilities of an M/M/s/m queue is suggested as a further approximation, since these probabilities of the general M/G/s/m queues are intractable. This leads to formula (10) of [9], from which the entries $W_q(NR)$ in Table 3 are calculated. From the numerical results presented in Table 3, one finds that in the case of work stations with limited storage capacity, as service time tends to constant, the performance measures (such as blocking probability π_m (for Poisson arrivals) and queueing delay W_q) differ significantly from those of exponential cases. A misused exponential assumption may cause some disastrous results (such as negative waiting times, see Table 3).

Example 2. Consider an $M/E_5/3/m$ work station, where traffic intensity is quite heavy: $\lambda = 5.61$. First let m = 6; performance measures are listed in the first row of Table 4. One finds the performance unsatisfactory, since the lost load (λ_{ls}) almost equals half of the offered load (λ_{of}). If the storage capacity is expanded to m = 9, while performance in general is hardly improved, delay increases significantly (row 2). However, if we

Table 1. $M/E_3/3/6$

λ	P_{id}	π_m	W_q
1.53	.7711 (.7679±.0559)	.0105 (.0105±.0049)	.0898 (.0975±.0202)
4.59	.0608 (.0654±.0088)	.3749 (.3673±.0106)	.6245 (.6341±.0165)

Table 2. $E_k/E_3/3/6$ $\lambda = 2.55$

k	P_{id}	π_m	L_s
4	.3845 (.3689±.0363)	.0468 (.0461±.0101)	2.9866 (3.0378±.0997)
5	.3838 (.3742±.0329)	.0439 (.0425±.0086)	2.9836 (3.0278±.0991)

Table 3. $M/E_k/3/6$ $\lambda = 2.55$

K	P_{id}	π_m	W_q	W_q(NR)
1	.4300 (.4358)	.1099 (.1087)	.3266 (.3221)	.3223
4	.3976 (.4165±.0335)	.0837 (.0787±.0109)	.2808 (.2782±.0212)	-.1036
∞	.3975 (.3856±.0351)	.0754 (.0654±.0101)	.2694 (.2737±.0182)	-2.455

Table 4. $M/E_5/3/m$

m	W_q	TH	λ_{1s}/TH	$\lambda_{1s}/\lambda_{of}$
6 $\lambda=5.61$	.7153 (.7327±.0045)	2.9479 (2.9525)	.9031 (.9001)	.4745 (.4737±.0031)
9 $\lambda=5.61$	1.6590 (1.6948±.0244)	2.9934 (2.9761)	.8741 (.8850)	.4664 (.4695±.0020)
6 $q=0.4$	.2956 (.2778±.0200)	2.7220 (2.7521)	.0676 (.0591±.0117)	.0633 (.0558)
6 $\lambda_{of}=2.906$	.3555 (.3731±.0224)	2.5019 (2.5372)	.1615 (.1444)	.1390 (.1262±.0118)

maintain the storage capacity (m = 6) and resort to flow control, then with a very simple strategy, i.e., whenever all the machines are occupied, λ is tuned down to λq (q = .4, say), performances can be improved significantly (at the price of a slight loss of throughput; see row 3). In this case, the effective offered load is calculated as $\lambda_{of} = 2.906$. However, if the station is offered with this equivalent load at a fixed rate, i.e., regardless of the state of the station, the performances (row 4) are much worse that those in the controlled input case.

4. CONCLUDING REMARKS

We have developed a diffusion approximation submodel which is able to accommodate the basic features of a work station in FMS. Using the model at the work station level, we are able to draw two important conclusions: (1) If the machining time is nearly deterministic, approximating it by an exponential time will significantly distort the evaluation of the station's performance. This is particularly so when the storage capacity is limited. (2) While increased storage capacity hardly helps improve performance, simple flow control strategies based on very limited amount of information can improve performance significantly.

Another input control strategy is to re-organize the arrival stream and input the jobs in batches. Our model can be extended to accommodate this feature; results will be reported elsewhere. A shortfall of our model is that when the service time distribution has a scv significantly bigger than 1, the accuracy is not adequate. We are currently investigating other approaches to address this problem.

References

[1] Bharucha-Reid, A.T. (1960). *Elements of the Theory of Markov Processes and Their Applications*. McGraw-Hill, New York.

[2] Buzacott, J.A. and Shanthikumar, J.G. (1980). "Models for Understanding Flexible Manufacturing Systems." *AIIE Trans.* *12*, 339-350.

[3] Buzacott, J.A. and Yao, D.D.W. (1982). "Flexible Manufacturing Systems: A Review of Models." Presented at TIMS/ORSA Detroit Meeting, 1982; copies available from the authors in the form of WP#82-07, Dept. of Industrial Engineering, University of Toronto.

[4] Cox, D.R. (1962). *Renewal Theory*. Methuen, London.

[5] Cox, D.R. and Miller, H.D. (1968). *The Theory of Stochastic Processes*. Wiley, New York.

[6] Feller, W. (1954). "Diffusion Processes in One Dimension." *Trans. Am. Math. Soc.* *77*, 1-31.

[7] Gelenbe, E. and Mitrani, I. (1980). *Analysis and Synthesis of Computer Systems*. Academic Press, London.

[8] Kimura, T. (1980). "Studies on Diffusion Approximation for Queueing Systems." Doctoral Dissertation, Kyoto University, Japan.

[9] Nozaki, S.A. and Ross, S.M. (1978). "Approximations in Finite-Capacity Multi-Server Queues with Poisson Arrivals." *J. Appl. Prob.* *15*, 826-834.

[10] Sevcik, K.C. et al. (1977). "Improving Approximations of Aggregated Queueing Network Subsystems." In Computer Performance, K.M. Chandy and M. Reiser (eds.), North-Holland.

[11] Shanthikumar, J.G. and Buzacott, J.A. (1980). "On the Approximations to the Single Server Queue." Int. J. Prod. Res. 18, 761-773.

[12] Shanthikumar, J.G. and Buzacott, J.A. (1981). "Open Queueing Network Model of Dynamic Job Shops." Int. J. Prod. Res. 19, 255-266.

[13] Solberg, J.J. (1977). "A Mathematical Model of Computerized Manufacturing Systems." In Proceedings: 4th International Conference on Production Research, Tokyo, August, 1977.

[14] Stecke, K.E. and Solberg, J.J. (1981). "The CMS Loading Problem." The Optimal Planning of Computerized Manufacturing Systems Report #20, School of Industrial Engineering, Purdue University, West Lafayette, Indiana.

[15] Whitt, W. (1982). "Approximating a Point Process by a Renewal Process, I: Two Basic Methods." Opns. Res. 30, 125-147.

Worst-Case Analyses of Self-Organizing Sequential Search Heuristics[1]

Jon Louis Bentley[2]
Catherine Cole
Department of Computer Science
Carnegie-Mellon University
Pittsburgh, Pennsylvania 15213

Abstract -- The performance of sequential search can be enhanced by the use of heuristics that move elements closer to the front of the list as they are found. Previous analyses have characterized the performance of such heuristics probabilistically. In this paper we show that the heuristics can also be analyzed in the worst-case sense, and that the relative merit of the heuristics under this analysis is different than in the probabilistic analyses.

1. Introduction

The performance of sequential search in an unsorted list can be enhanced by the use of heuristics that attempt to ensure that frequently accessed keys are near the front of the list.[3] The following three heuristics are representative of a larger class.

- Transpose. When the key is found, move it one closer to the front of the list by transposing it with the key immediately in front of it.
- Move-to-Front. When the key is found, move it to the front of the list (all other keys retain their relative order).
- Count. When the key is found, increment its count field (an integer that is initially zero). Keep the list sorted in decreasing order by count.

Note that the first two heuristics require no memory other than that for representing the lists, while the third heuristic requires an additional count field; the first two heuristics will therefore be called *memoryless*. Previous work (described in the next section) has shown that under various probabilistic assumptions, these heuristics can significantly reduce the time required by sequential search.

In this paper we will investigate the heuristics from a novel viewpoint: that of their *worst-case* performance rather than their *expected* performance. Such an analysis is trivial and uninteresting if we consider the worst-case cost of a single search. We will therefore count the worst-case number of comparisons made by the heuristics for any particular sequence of search keys, and show that for the Move-to-Front and Count heuristics, that number is at most twice the number of comparisons made when using the Optimal Static Ordering (defined in the next section) for the sequence of searches. This result immediately implies the strongest general theorem known for the expected time of the Move-to-

---

[1] This research was supported in part by the Office of Naval Research under contract N00014-76-C-0370 and in part by the National Science Foundation under an NSF Graduate Fellowship.

[2] Also with the Department of Mathematics.

[3] By the term list we refer only to its sequential nature; our results apply to lists implemented with arrays or with records and pointers.

Front heuristic. We also give a counter-example that shows that the Transpose heuristic has a very poor worst-case performance.

These analyses are of both theoretical and practical interest. This paper emphasizes the theoretical tools used in the worst-case analyses. Our results provide a simpler proof of a stronger theorem regarding an artifact that has been studied for well over a decade. Furthermore, the analyses use a simple but elegant bookkeeping technique of general interest. The practical contribution of this paper is not so much prescriptive as descriptive: practitioners have long used the Move-to-Front heuristic[4] even though theoretical analyses indicated that the Transpose heuristic was superior. Our analysis provides a metric under which Move-To-Front is superior to Transpose and thereby explains the actions of the practitioners.

This paper is organized in five primary sections. An overview of previous work can be found in Section 2. With that background, the new results are presented in Section 3. Section 4 describes the results of several empirical studies, and conclusions are offered in Section 5.

2. Previous Work

In this section we will survey previous results concerning the three heuristics. Only the more important results are are presented here; for further study, consult the reference list at the end of the paper.

The heuristics (or rules) that we study deal with searches for elements of a set of N keys stored in a list. A particular query is answered by performing a sequential search for the requested key and then reordering the list according to some search rule. A string of requests forms a *request sequence.* An important kind of request sequence independently chooses the i^{th} element with probability p_i according to the probability distribution $P=(p_1,p_2,\ldots,p_N)$. The *cost* of a search rule for such a distribution has been defined as the asymptotic expected search cost for a single key when the set is being reordered according to the rule; we will denote this cost by $A_R(P)$ for rule R. The Optimal Static Ordering for the set is one in which the keys are arranged in decreasing order of request probabilities and never reordered. While this is not necessarily optimal over all rules (because it is static, rather than dynamic), it is used as a basis for comparing the performance of the heuristics, and its cost will be denoted by $A_O(P)$. The heuristics have been studied under this asymptotic model since 1965; we present the significant results below. The Count heuristic is considered after the two memoryless heuristics.

The asymptotic expected search cost $A_M(P)$ under the Move-to-Front rule for the probability distribution P has been given by McCabe [1965], Burville and Kingman [1973], Knuth [1973], Hendricks [1976], Rivest [1976], and Bitner [1979]. It is known that for any distribution P, the cost $A_M(P)$ is at most twice the cost of the Optimal Static Ordering, $A_O(P)$. The asymptotic expected cost for Transpose, $A_T(P)$, was shown by Rivest [1976] to be less than or equal to $A_M(P)$; this bound is strict for all P except where all the nonzero probabilities are equal or when $N=2$.

---

[4] Some might question whether this work could ever have practical significance on the ground that if performance does matter, then non-linear data structures should be used for searching. That argument ignores two important facts: sequential structures are usually the proper structure to use in a first implementation of a module, and it is usually easier to incorporate these heuristics into existing code than to replace the code entirely. Furthermore, these heuristics arise in the analysis of chained hash tables in the eminently practical application of compiler symbol tables.

Rivest defined the *optimal permutation rule* to be one with least cost for all P and any initial ordering of the keys, and conjectured that Transpose is optimal. Gonnet, Munro, and Suwanda [1979] offered a modification of this definition. Yao (reported in Bitner [1976]) and Bitner [1982] have given distributions where Transpose is optimal over all rules, but Anderson, Nash, and Weber [1982] presented a counter-example to the conjecture by finding a rule that is better than Transpose for a specific distribution. Recent work has studied modifications of the heuristics and asymptotic costs for classes of probability distributions: see Gonnet, Munro and Suwanda [1979], Hendricks [1973], Bitner [1976, 1982], Kan and Ross [1981], and Tenenbaum and Nemes [1982]. A natural and interesting distribution is found under Zipf's Law; Knuth [1973] showed that the cost of Move-to-Front under this law is bounded by $2 \ln 2$ (about 1.386) times the cost of the Optimal Static Ordering. The highest ratio yet found is $\pi / 2$ (by Gonnet, Munro, and Suwanda [1979]), and the existence of a bound tighter than $2 \cdot A_O(P)$ remains an open problem

Measurements other than asymptotic cost have been considered. Bitner [1976, 1979] showed that while Transpose is asymptotically more efficient, Move-to-Front converges more quickly. Therefore, Move-to-Front is preferred when the number of requests is not large. He proposed a *hybrid rule*, which changes from Move-to-Front to Transpose when the number of requests falls in a certain range: from $\Theta(N)$ to $\Theta(N^2)$ requests is suggested for Zipf's Law. Bitner also discussed the *overwork* (the area between the cost curve and its asymptote) for the two rules, and presented distributions for which Move-to-Front performs much better than Transpose under this measure. Under Zipf's Law, the overwork is $O(N^2)$ for Move-to-Front and is $\Omega(N^3)$ for Transpose.

Rivest [1976] introduced a range of *move-ahead-k* heuristics, where a requested key is moved ahead k positions ($k=1$ corresponds to Transpose and $k=N-1$ corresponds to Move-to-Front), and simulated the asymptotic behavior of these heuristics for values of k from 1 to 6 and values of N from 3 to 12, for 5000 requests distributed by Zipf's Law. From these results, Bitner [1976] conjectured that for any two heuristics in this range, one will converge faster and the other will have lower asymptotic cost. Gonnet, Munro, and Suwanda [1979] later proved this. Tenenbaum [1978] performed similar tests for N between 3 and 230 and for 12,000 requests, with k from 1 to 7. His results indicate that for larger N and this number of requests a heuristic other than Transpose is more efficient.

The Count heuristic introduces a frequency count f_i of requests for the i^{th} key. Because of this extra information, Count has not been considered to be in the same class as the first two heuristics and so has received less attention. By the law of large numbers, if $p_i > p_j$, then the frequency f_i may be less than f_j for only a finite number of requests. The search cost under Count therefore asymptotically approaches that of the Optimal Static Ordering. Bitner [1976] showed that if P is not known beforehand, then Count is at all times optimal.

Various modifications of Count have been proposed to reduce the space needed to maintain the frequency counts. Bitner [1976, 1979] suggested that it is better to maintain the differences between frequencies of adjacent keys rather than their actual counts and also proposed a *limited-difference* rule, where the counts are left unchanged after some upper limit is reached. Other modifications have been suggested for use in combination with Move-to-Front or Transpose. Gonnet, Munro and Suwanda [1979], and Kan and Ross [1980] have examined *k-in-a-row* heuristics, where a key is moved only after it has been requested k times in a row, and Bitner [1976] has analyzed rules of the form *wait-c-and-move.* Lam, Sui, and Yu [1981] presented a scheme that was shown to be optimal over all heuristics that use frequency information.

3. Worst-Case Analyses

The primary results of the previous section can be summarized as follows: for any probability distribution P,

$$A_M(P) \leq 2 \cdot A_O(P),$$
$$A_T(P) \leq A_M(P), \text{ and}$$
$$A_C(P) = A_O(P).$$

These results all deal with the asymptotic expected cost of a single search when the queries are from a distribution P. In this section we will take an alternative view by considering the worst-case cost of performing all searches in a given sequence of queries S. When the list is being reordered by rule R, we will denote the total number of comparisons required for the sequence S by $C_R(S)$ (signifying the concrete cost as opposed to the asymptotic cost). The next two subsections will show that for any sequence S,

$$C_M(S) \leq 2 \cdot C_O(S), \text{ and}$$
$$C_C(S) \leq 2 \cdot C_O(S).$$

In Subsection 3.3 we will show that such a result does not hold for the Transpose heuristic by exhibiting a particular sequence with very poor performance under that rule.

3.1. Move-To-Front Heuristic

In this subsection we will show that for any particular sequence of requests S, the number of comparisons made by the Move-To-Front heuristic is never more than twice the number made under the Optimal Static Ordering.[5] To do this, we will reduce the problem to the case in which the request list contains just two distinct keys, analyze that simple case, and finally combine the facts to complete the proof.

The total number of comparisons made for a given request sequence can be divided into two kinds of comparisons: *intraword* comparisons (successfully) compare equal keys, and *interword* comparisons (unsuccessfully) compare unequal keys. For any sequence, the number of intraword comparisons is invariant under all heuristics. For the Move-To-Front heuristic, the total number of interword comparisons is the sum over all distinct pairs of keys of the number of interword comparisons made between each pair. Furthermore, for any sequence S and all pairs of keys A and B, the number of interword comparisons of A to B is exactly the number made in the subsequence of S consisting solely of A's and B's. We call this the *pairwise independence property* of the Move-To-Front heuristic; the number of comparisons made is dependent only on the relative ordering of the A's and B's in the sequence and is independent of other keys. The proof of the property is obvious: accessing an A will cause an (A,B) interword comparison if B is in front of A in the search list, which is true if and only if the last B was accessed more recently than the last A. The other keys in the request sequence do not affect this relationship.

We will now demonstrate the following fact.

---

[5] In this section we must redefine the Optimal Static Ordering to be the one where the keys are arranged by decreasing frequency of request, rather than by request probability.

Fact 1.

The total number of interword comparisons made by the Move-To-Front heuristic on a sequence of A's and B's is at most twice the number of interword comparisons made by the Optimal Static Ordering applied to the same sequence.

To prove the fact we will assume that the sequence S consists of m A's and n B's, where (without loss of generality), $m \leq n$. Note that under the Optimal Static Ordering, a total of m interword comparisons will be made (because the search list is always in the order B A, and so only requests for A will cause an interword comparison). Under the Move-To-Front rule, an interword comparison will be made whenever the sequence changes from an A to a B or from a B to an A. The total number of such changes possible is just twice the number of occurrences of A's (for each change involves an A, and each A can be involved in at most two changes). We therefore know that the total number of comparisons made by Move-To-Front is at most $2m$. Fact 1 follows immediately.

We are now ready to prove the key fact of this subsection.

Fact 2.

For any sequence S, $C_M(S) \leq 2 \cdot C_O(S)$.

We will prove this by simple algebra on the relations

$$C_M(S) = Intra(S) + Inter_M(S) \text{ and}$$
$$C_O(S) = Intra(S) + Inter_O(S),$$

where *Intra* and $Inter_R$ refer to the total number of comparisons of each type made by rule R. By Fact 1 we know that each pair of keys satisfies the factor of two inequality; summing over that inequality for all distinct pairs gives

$$Inter_M(S) \leq 2 \cdot Inter_O(S),$$

for any sequence S. Combining this inequality with the above definition gives Fact 2.

Note that the factor of two in Fact 2 cannot be tightened. The request sequence

$$\text{A B C D (D C B A)}^m$$

has $C_O(S) \sim 2m$ but $C_M(S) \sim 4m$.

Knuth [1973, Exercise 6.1-11] shows that $A_M(P) \leq 2 \cdot A_O(P)$, for any distribution P; that exercise is rated M30 in his scheme (implying it is mathematically oriented and may require over two hours' work to solve). Fact 2 allows us to prove that result easily: we let the sequence S be an arbitrarily long sequence chosen from the distribution P. By Fact 2, the cost of Move-To-Front applied to that sequence is at most twice the cost of the Optimal Static Order for the sequence. But the cost of the Optimal Static Order for that particular sequence is certainly less than or equal to the cost of the Optimal Static Order for the corresponding distribution, so the asymptotic result is established.

3.2. Count Heuristic

In this subsection we will show that the cost of the Count heuristic on any particular sequence is at most twice the cost of the Optimal Static Ordering. The flow of this subsection is exactly the same as the previous subsection, so we will merely sketch the proof.

The first fact we have to establish is that the Count heuristic has the pairwise independence property: for any sequence S, the number of interword comparisons of A and B is exactly the number made in the subsequence of S consisting solely of A's and B's. This is easily proved: A will precede B in the search list only if A has a greater count than B, and the values of the counts are not affected by other keys. As in the Move-To-Front heuristic, this pairwise independence allows us to focus on two-element sequences. We therefore prove the following fact.

Fact 3.

The total number of interword comparisons made by the Count heuristic on a sequence of A's and B's is at most twice the number of interword comparisons made by the Optimal Static Ordering applied to the same sequence.

To prove this fact, we again assume that the sequence S consists solely of m A's and n B's, with $m \leq n$. Under the Count heuristic, an interword comparison is made every time the key in the rear of the search list is requested; at that time, its count field is incremented. Now, A can have its count field incremented while it is in the rear at most m times (because it is requested m times). Furthermore, B can have its count field incremented while it is in the rear at most m times (because after that it has been requested more than m times and can no longer lie in the rear). The number of interword comparisons, then, is bounded by m (for the A's) plus m (for the B's), or $2m$, which is twice the number of comparisons made by the Optimal Static Ordering.

The key fact of this subsection follows from the same kind of reasoning used to establish Fact 2 of the previous subsection.

Fact 4.

For any sequence S, $C_C(S) \leq 2 \cdot C_O(S)$.

Again, the reasoning involves summing over the factor of two inequality. As before, the factor of two cannot be tightened.

3.3. Transpose Heuristic

In this subsection we will demonstrate that the worst-case ratio of the performance of Transpose to that of the Optimal Static Ordering cannot be bounded by any constant. This is easily observed if we consider the request sequence

$$\text{A B C D E (E D)}^k$$

After the first five elements are stored by Transpose, the sequence of (E D) request pairs will cause those two elements to swap position at the back of the list, and neither will advance. The average cost of a search in this sequence will therefore approach 5, while under the optimal static ordering 1.5 comparisons would suffice. For increasing k, this example gives

$$C_T(S) > 3.33C_O(S).$$

The constant 3.33 can be increased to $\sim 2k/3$ by increasing the length of the "filler" sequence preceding the "active" pair to $k-2$. Note that this counterexample exploits the fact that the Transpose heuristic does not have the pairwise independence property: the relative order of any two keys depends not only on the request sequence but also on whether the keys are adjacent in the search list.

4. Empirical Results

The theoretical analyses in the previous sections are by no means unanimous in their evaluation of the heuristics. To gain further insight into the behavior of the heuristics, we used each to perform word counts on a variety of files; that is, the words in each file served as a request sequence. As each word (defined to be an alphanumeric string delimited by spaces or punctuation marks) was requested, a linear search of the key list was performed, the count field for the key was incremented, and the list was reordered according to the appropriate rule. Each started with an initially empty key list; at the first request for a word, the list was searched to the end to determine its absence and then the reordering occured as if the element had been found in the (new) last position. This application clearly suggests that the Count heuristic be used, since the frequency counts must be stored anyway. This type of input may be used in other contexts, however, so the test serves as a good indicator of the behavior of the heuristics under natural conditions.

The total number of interword comparisons required by each heuristic for each file was reported as the performance measure and was compared to the corresponding cost for the Optimal Static Ordering. A table comparing the performances of the three heuristics appears later in this section (the best performance for each file is underlined), and the results are summarized below. While Transpose never performed as badly as our results showed it might, in every case it required significantly more comparisons than Count and Move-to-Front.

The files were obtained from user accounts and an on-line documentation system, and were grouped into three classes: Pascal files, Scribe files, and Text files. The characteristics of the classes vary, so we consider the results for each separately.

The four Pascal files tested contained between 100 and 181 distinct words (corresponding to the length of the key list), which were requested a total of 431 to 1456 times (corresponding to the length of the request sequence). These were the *low-vocabulary* files: the ratio of the length of the calling sequence to the length of the key list was high. This ratio should deliver results closer to asymptotic than for the other file types, and so Transpose might be expected to perform best here. However, the empirical results for the Pascal files were striking: Move-to-Front and Count performed dramatically better than Transpose, and in two cases, Move-to-front required fewer comparisons than the Optimal Static Ordering! The high *locality* present in source code accounts for this performance: infrequently used words such as **integer** appear in groups rather than being uniformly distributed throughout the file. Where a request for such a word would require a long search under the Optimal Static Ordering, the search under Move-to-Front would be short after the first request, since the key would then be at the beginning of the list. For Transpose, however, the requested element doesn't have time to drift towards the front, and so the search remains expensive. This phenomenon of locality (in such constructs as **Total := Total + 1, end; end; end**, and **var** declarations) enhances the performance of Move-to-Front considerably.

Scribe files are text files embedded with keyword instructions to the Scribe document production system. The files contained between 49 and 289 words which were requested from 145 to 688 times. These are *high-vocabulary* files (the average ratio of request sequence length to key list length was small), which indicates that Transpose may not have a chance to approach its asymptote. However, the results were again suprising: while Transpose still required more comparisons than the other two heuristics, its performance was best for this class of files. Move-to-Front used fewest comparisons for all but one of these files and Count's performance was close to that of Move-to-Front. In one case, Count required the fewest comparisons.

The Text files included the text of the Constitution of the United States, the script to *The Rocky Horror Picture Show*, and excerpts from an on-line documentation system. Two of the files were very large, containing 1147 and 1590 distinct words that were requested 7482 and 7654 times, respectively. For these very large files, the Count heuristic outperformed both Move-to-Front and Transpose. Move-to-Front still required fewer comparisons than Transpose, however, and Move-to-Front used the fewest comparisons for the other text files.

	Distinct Words	Total Requests	Optimal Static	Move-to Front	Count	Transpose
Pascal Files						
P 1	100	480	13,210	11,753	15,919	19,408
P 2	107	431	11,307	11,043	13,578	16,776
P 3	117	1,176	21,210	21,349	24,255	35,909
P 4	181	1,456	44,820	45,724	52,001	69,035
Scribe Files						
S 1	49	145	1,789	2,220	2,047	2,438
S 2	174	306	17,013	18,612	19,981	22,914
S 3	238	523	34,117	38,351	40,696	44,254
S 4	289	688	50,383	55,824	60,519	75,749
Text Files[6]						
T 1	471	1,888	175,639	197,227	209,971	278,310
T 2	498	1,515	170,976	180,754	205,477	243,448
T 3	1,147	7,482	1,075,289	1,305,639	1,242,793	1,531,861
T 4	1,590	7,654	1,779,808	2,149,440	2,048,479	2,678,437

Table 1. Interword Comparisons

The empirical results indicate that asymptotic analyses do not accurately predict the behavior of the heuristics under natural conditions. Certainly, the distribution and size of the request sequence seem be less significant than the ordering of the requests. Empirical results for the Pascal files would be less dramatic if (localized) keywords were treated differently from identifiers, as a compiler might do. However, if sequential search were used by an interpreter for identifier lookup, the presence of *dynamic locality*, for example in requests for loop variables, would argue strongly for the use of Move-to-Front.[7] This phenomenon is not restricted to source code; for example, the word "president" appears with high locality in the U.S. Constitution. In most written prose, locality of subject (and therefore of subject words) determines paragraph construction. Move-to-Front is clearly able to take advantage of this characteristic. Indeed, such a phenomenon as "pairwise-localty" (the appearance of word pairs or two-word phrases such as "vice-president" or A[I]) might hamper the performance of Transpose; if two such words have the misfortune to be adjacent in the key list, they will contend with each other rather than drifting to the front as they should.

---

[6] The third and fourth files are the texts of the U. S. Constitution and *The Rocky Horror Picture Show*, respectively.

[7] The widely-used Microsoft BASIC interpreter uses the heuristic of storing symbols in its run-time (linear) symbol table in the order in which they were first seen. One of the authors (JLB) once reduced the run time of a BASIC program under such an interpreter from fourteen hours to seven hours simply by referring to each "hot" variable once in a dummy statement at the front of the program. The use of the Move-To-Front heuristic in such an interpreter would probably substantially decrease the run time of many BASIC programs.

The Count heuristic performed fairly well in most cases, staying close to Move-to-Front. This is the only heuristic that leaves the list ordered by frequency of request, which is useful if these frequencies are to be reported. However, the interword comparisons reported for Count do not include the additional comparisons required to "sift-down" an element in the key list when its count field is big enough to require reordering. If these comparisons are counted, then Move-to-Front requires the fewest number of comparisons in every case. Also, the extra fields needed to maintain the frequency counts may make the space requirements prohibitive if the keys themselves are small.

5. Conclusions

The conclusions of this paper are clear: when a self-organizing sequential search is appropriate in an application, the Count and (especially) the Move-To-Front heuristics should be considered for implementation. Although previous probabilistic analyses showed that Transpose is superior to Move-To-Front under some measures, both our worst-case analyses and our empirical results show contexts under which the opposite is true.

The results in this paper could be extended in a number of ways. The worst-case analysis of algorithms previously analyzed only for their expected performance is an interesting open problem. To predict more accurately the behavior of the heuristics on input like that described in the previous section, it would be helpful to have theoretical tools for describing the locality present in the input. Also, the proof techniques that we presented could perhaps be used to study other self-modifying structures in the worst-case sense.

Acknowledgements

The helpful comments of Jim Saxe and Bruce Weide are gratefully acknowledged.

References

Anderson, E. J., P. Nash, and R. R. Weber [1982] A counterexample to a conjecture on optimal list ordering. *J. Appl. Prob.*, to appear.

Bitner, J. R. [1976]. *Heuristics that dynamically alter data structures to reduce their access time.* Ph.D Thesis, University of Illinois, Urbana-Champaign, IL.

Bitner, J. R. [1979]. Heuristics that dynamically organize data structures. *SIAM J. Comp. 8*, 1 (February 1979), 82-110.

Bitner, J. R. [1982] Two results on self-organizing data structures. Department of Computer Science, University of Texas at Austin, Austin, TX, TR-189 (January 1982).

Burville, P. J. and J. F. C. Kingman [1973]. On a model for storage and search. *J. Appl. Prob. 10*, (September 1973), 697-701.

Gonnet, G., J. I. Munro, and H. Suwanda [1979]. Toward self-organizing sequential search heuristics. *Proc. 20th IEEE Symp. Foundations Computer Science*, 169-174.

Hendricks, W. J. [1972]. The stationary distribution of an interesting Markov chain. *J. Appl. Prob. 9*, (March 1972), 231-233.

Hendricks, W. J. [1973]. An extension of a theorem concerning an interesting Markov chain. *J. Appl. Prob. 10,* (December 1973), 886-890.

Hendricks, W. J. [1976]. An account of self-organizing systems. *SIAM J. Comp. 5,* 4 (December 1976), 715-723.

Kan, Y. C. and S. M. Ross [1980]. Optimal list order under partial memory constraints. *J. Appl. Prob. 17,* (December 1980), 1004-1015.

Knuth, D. E. [1973]. *The Art of Computer Programming, Vol 3: Sorting and Searching.* Addison-Wesley, Reading, MA.

Lam, K., M. K. Sui, and C. T. Yu [1981]. A generalized counter scheme. *Theoretical Comp. Sci. 16,* (December 1981), 271-278.

McCabe, J. [1965]. On serial files with relocatable records. *Oper. Res. 12,* (July 1965), 609-618.

Rivest, R. [1976]. On self-organizing sequential search heuristics. *CACM 19,* 2 (February 1976), 63-67.

Schay, G. Jr., and F. W. Dauer [1967]. A probabilistic model of a self-organizing file system. *SIAM J. Appl. Math. 15,* (February 1967), 874-888.

Tenenbaum, A. [1978]. Simulations of dynamic sequential search algorithms. *CACM 21,* 9 (September 1978), 790-791.

Tenenbaum, A. and R. M. Nemes [1982]. Two spectra of self-organizing sequential search algorithms. *SIAM J. Comp.,* to appear.

Veinott, A. F. [1965]. Optimal policy in a dynamic, single product, nonstationary inventory mode with several demand classes. *Oper. Res. 13,* 761-778.

A NEW TIME COMPLEXITY FOR ORTHOGONAL RANGE QUERIES
(An Extended Abstract)

Dan E. Willard
Bell Laboratories
Holmdel, New Jersey 07733

GOAL

This article introduces several new algorithmic techniques which make possible data structures occupying space $O(N \log^{k-1} N)$ and supporting a time complexity $O(\log^{k-1} N)$ for the most common types of k-dimensional orthogonal range queries ($k \geq 2$). This time-complexity is a factor log N improvement over ten previous authors, implying a factor 20 improvement when memory space is doubled and a full factor of 3 improvement if it increases by only 2% in practical applications where $k = 2$.

1. INTRODUCTION

A k-dimensional orthogonal range query will be defined as a request for the records satisfying a predicate condition similar to

$$a_1 < \text{KEY.1} < b_1 \ \& \ a_2 < \text{KEY.2} < b_2 \ \& \ \ldots \ a_k < \text{KEY.k} < b_k . \qquad (1)$$

Throughout this paper q will denote a query similar to Eq. 1 and F_q the number of records that satisfy this query. A query-algorithm q will be said to have a worst-case aggregate query complexity $O(g(N))$ iff it uses this time to calculate an aggregate; it will be defined to have a worst-case locate complexity $O(g(N))$ iff it uses time $O(g(N) + F_q)$ to find and print the list of F_q records satisfying q. This paper will propose one static data structure, Ta(k), which provides worst-case aggregate and locate complexities $O(\log^{k-1} N)$ in space $O(N \log^{k-1} N)$, and several possible modifications of this structure for a dynamic environment.

The precursor to our proposal has been discussed in [2,3,5,8,13,15,17]; we will call it a first-generation k-fold tree and denote it T1(k). These trees can be constructed in time $O(N \log^{k-1} N)$ and allow one to calculate a batch of N ECDF statistics in the same time [3,5]. Willard and Leuker [22] have shown how to perform insertions and deletions in worst-case time $O(\log^{k} N)$ under this data structure, and one of the several consequences from Fredman's formalism [11,12] is the optimality of this result for the version of the orthogonal range query problem which calculates aggregates in a dynamic environment. Several authors in various contexts [2,3,5,8, 13,15,17] have noticed that first-generation k-fold trees have locate and aggregate complexities $O(\log^{k} N)$ in both the best and worst-cases. Our basic proposal, called second-generation k-fold trees, occupy the same space, $O(N \log^{k-1} N)$, as first-generation trees but reduce the preceeding time-complexity by a factor of log N in both a static environment and in most dynamic environments not precluded by Fredman's lower bound [11,12].

Our research goal should not be confused with [4]; the latter article showed how a time complexity O(log N) was possible in a memory space much larger than $O(N \log^{k-1} N)$. As a rule, our data structures have a better space but worse time complexity than [4]. Our proposal is especially attractive when k = 2; then it matches the time O(log N) of [4] while using significantly less memory.

Five different variations of our proposal, denoted Ta(k), Te(k), Td(k), Tf(k), and Tb(k), will be presented in this extended abstract. Each will occupy memory space $O(N \log^{k-1} N)$. Ta(k) is the static tree described in the first paragraph of this article. Te(k) is a modification of Ta(k) which has the same locate-retrieval complexity, an expected time $O(\log^{k-1} N)$ for insertions and deletions, but no capacity for calculating aggregates. Tf(k) is a further modification of Te(k) which guarantees a worst-case cost $O(N \log^{k} N)$ for a sequence of N insertion commands when absolutely no deletions are present. Td(k) is an alternate modification of Te(k) which has an expected time $O(\log^{k-1} N)$ for insertions, deletions, and locate-retrievals and a worst-case bound $O(\log^{k} N)$ for these three operations. Tb(k) is a data structure with worst-case complexity $O(\log^{k-1/2} N)$ for insertions, deletions, and locate-retrievals.

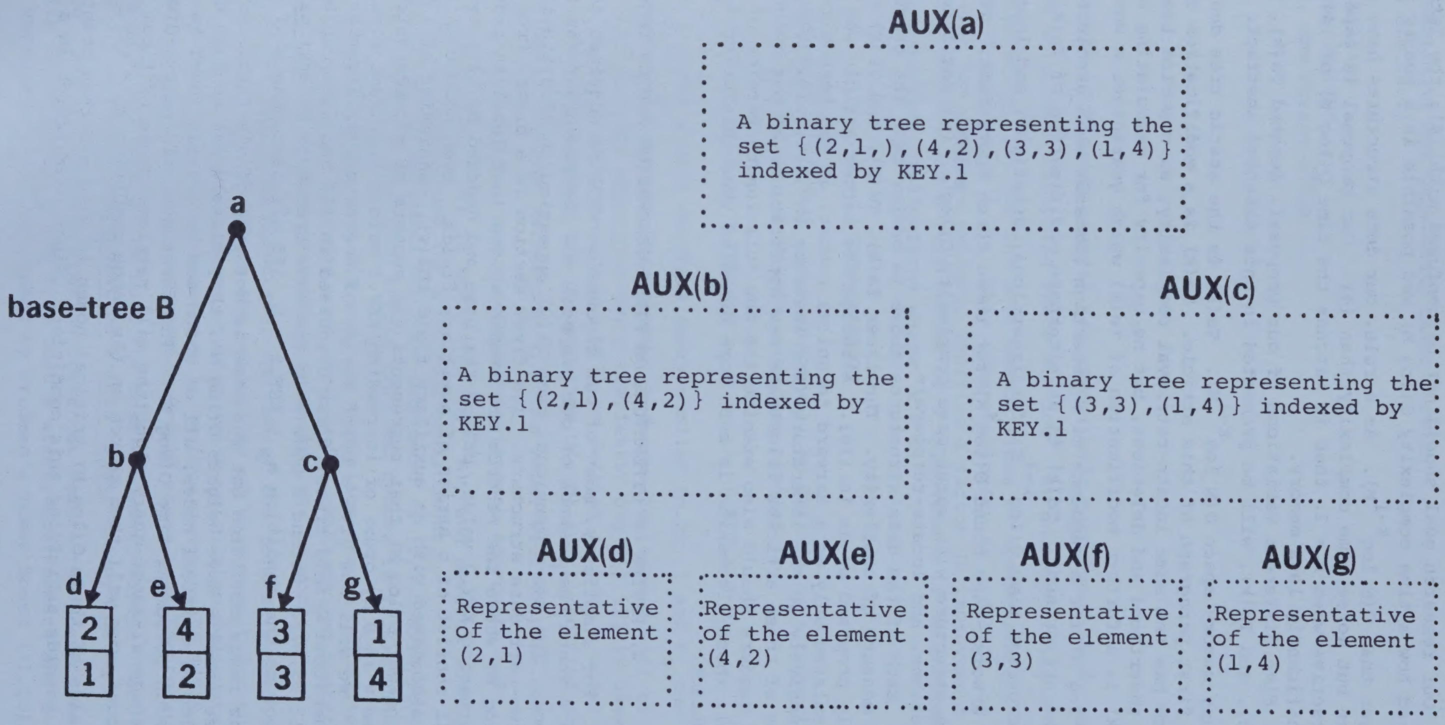
Each of the data structures above is better than the others by at least one measure of complexity. The trees Ta(k), Td(k), and Te(k) were originally proposed by us in [18]. Although the latter article was previously circulated only as a Harvard technical report, it has been cited quite extensively in the literature (for instance in [3,6,9,10,12,14,16,21,22,23]). Some of these articles illustrate new applications of our method. Interested readers should also examine the new full-length version of our paper [20], whose discussion is much more detailed than this brief extended abstract.

2. REVIEW OF LITERATURE ON FIRST-GENERATION K-FOLD TREES

For simplicity, most of our discussion will be confined to dimension k = 2 where the essence of our proposal and its precursor can be understood. The two-dimensional tree T1(2) appearing in [2,3,5,8,13,15,17] is a two-part data structure. Its first section is a binary tree of height O(log N) where the records are stored at the leaf-level in order of increasing KEY.2 value, called the base and denoted B. Its second part will consist of a series of auxillary fields. Each node v in the base will be associated with an auxillary field AUX(v), consisting of a binary tree of height O(log N) that represents the subsets of records which descends from v in B by order of increasing KEY.1 value, i.e., see Figure 1.

We will now give a brief review of the retrieval algorithm for first-generation 2-fold trees, first discussed in [5] and also employed in [2,3, 8,13,15,17]. Define a node v in the base tree B to be critical with respect to the condition $a_2 < \text{KEY.2} < b_2$ iff all descendants of v satisfy this range condition but the same is not true of v's father. Since the base trees B have heights O(log N), they clearly can contain no more than O(log N) critical nodes, all of which can be clearly found in a single search consuming time O(log N). The first step of the two-dimensional orthogonal-range-query algorithm will retrieve these critical nodes. The second step will then search on the range condition $a_1 < \text{KEY.1} < b_1$ the fields of these O(log N) critical nodes to find and then print the list of records satisfying this condition. Since each search in a binary tree

FIGURE 1: A first-generation 2-fold tree representing the set: {(2,1),(4,2),(3,3),(1,4)}



AUX(a)
A binary tree representing the set {(2,1,),(4,2),(3,3),(1,4)} indexed by KEY.1
base-tree B
a
b
c
d
e
f
g
2
1
4
2
3
3
1
4
AUX(b)
A binary tree representing the set {(2,1),(4,2)} indexed by KEY.1
AUX(c)
A binary tree representing the set {(3,3),(1,4)} indexed by KEY.1
AUX(d)
Representative of the element (2,1)
AUX(e)
Representative of the element (4,2)
AUX(f)
Representative of the element (3,3)
AUX(g)
Representative of the element (1,4)

requires logarithmic time to initiate its retrieval and since our orthogonal-range-query algorithm searches O(log N) distinct auxiliary fields, it it not hard to see that this procedure uses locate-time $O(\log^2 N)$ both in its best and worst cases.

3. OUR BASIC RESULT FOR DIMENSION K=2

We claim that by introducing a new type of pointer, one can attain the more efficient time O(log N) in the same space $O(N \log^{k-1} N)$ as before. Some notation is useful to help define our new concept. Note that the definition of first-generation 2-fold trees requires that each record R, descending from the node v in the base, must contain some representative of R in AUX(v). This representative will be called R's entry in AUX(v) and will be denoted typically by X or Y, with X.i and Y.i denoting the i-th keys of these entries. Also for each entry X belonging to AUX(v), its left-down-son will be defined as that entry Y in the auxiliary field of v's left son with the smallest Y.1 value satisfying Y.1 $\geq$ X.1. The right-down-son of X will be defined as the analogous entry satisfying Y.1 $\geq$ X.1 in the auxiliary field of v's right son. The second-generation 2-fold tree Te(2) will be defined to be a data structure identical to a first-generation 2-fold tree except for the following three changes:

i) Each entry X will contain two special pointers to its left- and right-down-sons; these will be called downpointers.

ii) The records in each auxiliary field will be arranged as a doubly linked list sorted by KEY.1 rather than in the binary-tree format of first-generation trees. (A Special entry serving as a termination mark will be placed at the end of each list, marking the position that would be occupied by the Key +∞. This termination mark should be treated as a normal entry when constructing the downpointers linking together the auxiliary fields.)

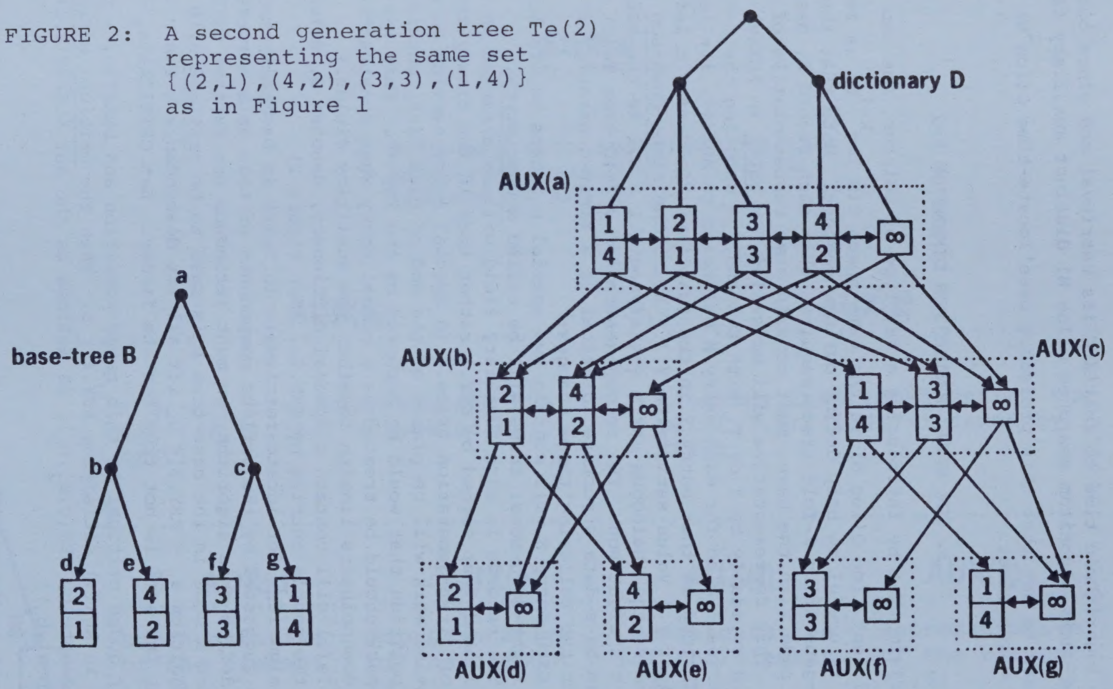
iii) Te(2) will contain a special dictionary, denoted D, which indexes the root's entries by KEY.1. (See Figure 2)

The algorithm for locate-retrievals in Te(2) is basically the natural algorithm suggested by the various components of this data structure. In order to define this algorithm, we must introduce one further definition. Recall that a node in the base-tree B is said to be critical with respect to the condition $a_2 <$ KEY.2 $\leq b_2$ iff all its descendants satisfy this condition but the same is not true for its father. Let CRITICAL(a_2,b_2) denote the set of nodes critical to this range condition and LEAST(v,c) the smallest entry in AUX(v) satisfying KEY.1 $\geq$ c. Then the critical entry set, which we denote by CES(c,a_2,b_2), is defined as the set {LEAST(v,c) | v $\in$ CRITICAL(a_2,b_2)}.

Theorem 1: For any three constants c, a_2, b_2, it is possible to retrieve CES(c,a_2,b_2) from Te(2) in time O(log N).

Proof Sketch: Let X belong to AUX(root). Define the entry-tree of X, denoted t(X), to be a binary tree whose root is X, whose depth-1 nodes are the left- and right-down-sons of X, whose depth-2 nodes are the sons of these sons, and so on. Our algorithm for retrieving a critical entry set will consist of three steps. Its first step will search the dictionary D to find that entry X belonging to AUX(root) which has the smallest X.1 value satisfying X.1 $\geq$ c. The second step will walk the subtree inside the base-tree B whose nodes are either critical with respect to the condition $a_2 <$ KEY.2 $\leq b_2$ or are the ancestors of critical nodes. The third step will

FIGURE 2: A second generation tree Te(2) representing the same set {(2,1),(4,2),(3,3),(1,4)} as in Figure 1


dictionary D
AUX(a)
1
4
2
1
3
3
4
2
∞
AUX(b)
2
1
4
2
∞
AUX(c)
1
4
3
3
∞
base-tree B
a
b
c
d
e
f
g
2
1
4
2
3
3
1
4
AUX(d)
AUX(e)
AUX(f)
AUX(g)

walk the isomorphic image of the subtree s lying inside the entry-tree t(X), using the downpointers to locate each node that it traverses. Each element from CES(c,a_2,b_2) is visited at this time because these elements are the bottom-level nodes of the subtree traversed. Since each of the three steps run in time O(log N), it follows that this is the complexity for retrieving critical entry sets. Q.E.D.

Theorem 1 suggests a natural algorithm for performing the two-dimensional orthogonal-range-query

$$Q = \{a_1 < \text{KEY.1} \leq b_1 \quad a_2 < \text{KEY.2} \leq b_3\}. \tag{2}$$

The first step of this algorithm will retrieve the critical entry set CES(c,a_2,b_2) by using the procedure from Theorem 1. The second step will take each entry X from CES(a_1,a_2,b_2) and walk in a rightward direction down the doubly linked list in its auxiliary field until it has encountered and printed all the entries Y in this field with $Y.1 \leq b_1$. Since the doubly linked lists are ordered by KEY.1, an easy analysis shows that this procedure performs two-dimensional orthogonal range queries in worst-case locate time O(log N).

4. SEVERAL EXTENSIONS AND IMPLICATIONS OF OUR BASIC RESULT

These proposals were first discussed by us in [18] and they are explained in greater detail in both [18] and [20]. These reports also illustrate extensions of the algorithm from the previous section. We will now briefly outline these further results, concentrating mostly on the case where k = 2.

Suppose that each record R contains a special field f(R) and that we wish to calculate the sum of the values f(R) of the records satisfying the query Q given in Eq. (2). Then it is desirable to employ a modified second-generation 2-fold tree, denoted Ta(2), whose every entry contains an aggregate field, denoted A(X), indicating the sum of the entries lying to its left in the auxiliary field. Clearly

$$\sum_{R \in Q} f(R) = \sum_{X \in \text{CES}(b_1,a_2,b_2)} A(X) \quad - \sum_{X \in \text{CES}(a_1,a_2,b_2)} A(X) \quad . \tag{3}$$

Eq. 3 implies aggregates can be calculated in time O(log N) under Ta(2), since Theorem 1 showed the critical entry sets on its right side can be retrieved in this time.

The tree Ta(2) does not have a good insertion-deletion complexity, and Fredman's lower bound shows no data structure can do better than $\Omega(\log^2 N)$ for the sum of its update and two-dimensional aggregate-retrieval complexities. However, [11,12] do not impose a similar lower bound for the sum of a data structure's locate-retrieval and update time, and [20] shows Te(2) has an update-complexity O(log N) at least in the expected case. The rest of this section will describe modifications of Te(2) which also control insertion-deletion costs in the worst case.

For simplicity we first assume that KEY.1 and KEY.2 are positive integers $\leq$ M and that $(\log M) \in O(\log N)$. Then the base B of our k-fold trees can be organized as a binary trie of height $\lceil \log M \rceil$. Since tries never need AVL-like rebalancing operations to maintain logarithmic height, our insertion-deletion algorithm will not need the methods which Willard and Lueker [22] applied to processing noninteger Keys.

In this case, the tree Te(2) will have good expected complexities for insertions and deletions by a simple analysis in [20]; however, it will not control worst-case costs because $\Omega(N)$ downpointers may require modification after the insertion or deletion of a single record. This difficulty can be resolved with a slightly modified tree, called Td(2), which has no downpointers, and instead represents the set of entries X with a common down-son Y as a 2-3 tree with root Y, and leaves corresponding to these entries X. The retrieval algorithm for Td(2) will be the same as Te(2) except that it will use the 2-3 trees rather than downpointers to find a node's down-son. The full length version of our report shows these trees have expected height O(1) and worst-case height O(log N). Combined with the results from the last section and [1]'s algorithm for merging and splitting 2-3 trees, these facts imply Td(2) has an expected complexity O(log N) and a worst-complexity $O(\log^2 N)$ for insertions, deletions, and locate-retrievals.

A second proposed tree, called Tc(2), consists of a data structure which uses q-fast tries [19] to form the auxiliary fields. The latter data structure has a complexity $O(\sqrt{\log M})$ for insertions, deletions, and one-dimensional retrievals when the Keys are positive integers $\leq M$. In the context of Tc(2) and our assumption $(\log M) \in O(\log N)$, this data structure provides worst-case complexities $O(\log^{3/2} N)$ for insertions, deletions, and two-dimensional locate retrievals.

Sometimes, a dynamic environment will include only insertion and retrieval commands. If deletion commands are absent, it will be desirable to use a tree, called Tf(2), defined as follows: Let S(X) denote the set of entries Y whose downson is X, and LS(X) and GS(X) the least and greatest elements of S(X). Define Tf(2) to be identical to Te(2) except that each entry X will contain two additional pointers to LS(X) and GS(X). We can apply these two additional pointers to guarantee a sequence of N insertion commands applied to an initially empty copy of Tf(2) will consume no more time than $O(N \log^2 N)$.

Proof Sketch: Suppose our insertion algorithm wishes to place a new record R immediately to the left of an older entry X in some auxiliary field. Let S_1 and S_2 denote the cardinalities of S(X) before and after the insertion of R. The record R can be inserted into the doubly-linked list in one of two ways. It can either be placed into a new entry-position between X and X's former predecessor, or it can be put into X's actual position with X being moved into a new position between its former position and its successor. The number of downpointers needing modification after these operations is $S_1 - S_2$ and S_2, respectively. The pointers LS(X) and GS(X) make it possible to execute insertions in such runtimes. Therefore, an insertion will have a cost $\mathrm{MIN}(S_1 - S_2, S_2)$ if we execute the less expensive alternative.

Let $|S(X)|$ denote the cardinality of S(X) and E the set of entries in Tf(2). Consider the accounting function

$$A = \sum_{X \in E} |S(X)| \cdot \log |S(X)|.$$

Since the insertion of a record causes log N entries to be inserted, one command will increase A by no more than $O(\log^2 N)$. The previous paragraph implies a readjustment involving j downpointers will decrease A by $\Omega(j)$. Since A is initially zero, the total readjustment cost for a sequence of N commands must be $O(N \log^2 N)$, by A's growth and readjustment properties. Other costs are also encountered during insertion operations, but these are shown to satisfy an even tighter bound O(log N) per operation in the full-length version of our paper [20]. Q.E.D.

One further refinement of our method should very definitely be outlined. A possible disadvantage of the trees Ta(2), Td(2), Te(2), and Tf(2) is that they occupy more memory space than the first generation tree Tl(2). This disadvantage arises because each entry in the second generation tree uses two extra units of memory on account of its downpointers, causing them to have larger coefficients associated with their O(N log N) space than first generation trees. For any $\varepsilon > 0$, these difficulties can be overcome with modified data structures whose time exceeds Ta(2), Td(2), Te(2), and Tf(2) by a factor of only log $(1/\varepsilon)$ and whose space exceeds Tl(2) by a factor of only $1+\varepsilon$. This result leads to some surprisingly practical improvements over the previous literature. For instance, if $\varepsilon = 2^{-6}$ and $N = 2^{18}$, then the revision will be three times faster than Tl(2) but occupy only 1.6% extra memory. A nine-fold improvement in 25% extra space or six-fold improvement in 12.5% excess space is possible if we set ε equal to 2^{-2} and 2^{-3}, respectively. The verification for Te(2) of these claims appears below:

Proof Sketch: Let M_X denote the number of entries Y in AUX(v) satisfying Y.1 > X.1, and say X is distinguished relative to a fixed constant J in this auxiliary field if J divides M_X. The proposed modification of Te(2) will divide its auxiliary fields into two parts. The first section will include only distinguished entries and connect them with downpointers similar to Te(2). The second section of AUX(v) will be an ordered array enumerating all its elements. An entry X in the first section will have a pointer, denoted FINGER(X), to its counterpart in the array.

If v is a critical node with respect to the query in Eq. 2, then let DLEAST(v,c) denote the smallest distinguished entry with X.1 > c. The set of entries {DLEAST(v,c) where v critical} can be found in time O(log N) by using Theorem 1's algorithm. Each element LEAST(v,c) can then be found in log J additions time if one makes a binary search between the addresses FINGER(X) - J+1 and FINGER(X). Therefore a factor log J separates the locate time of Te(2) and its revision. The time and space claims of our theorem follow by setting $J \cong 1/\varepsilon$. Q.E.D.

Comment 1: Our revision of Te(2) is actually slightly better than outlined above. Its worst-case locate component exceeds Te(2) by a factor $|\log \varepsilon|$, but its expected locate component will compare far better. The reason is that the binary search above sought to find LEAST(v,c) while any entry satisfying the query Q could be used to initiate the walk through AUX(v).

Comment 2: The method above generalizes to a dynamic environment if one treats FINGER(X) as a root of a 2-3 tree which contains between J and 2J entries. The split and merge algorithm of [1] should be applied to guarantee these trees always have cardinality between J and 2J.

5. FURTHER GENERALIZATIONS

All the results above can be generalized for the cases where the Keys are real numbers. The modification of the trees Td(2), Te(2), and Tf(2) for real numbers is easier than Tc(2) because the previous section did not actually need KEY.1 to be an integer in its treatment of Td(2), Te(2), and Tf(2). The bounded balance method of Willard and Lueker [22] will provide the machinery for handling real numbers in KEY.2 for these three trees.

The treatment of Tc(2) is more difficult because the tree's q-fast tries require that KEY.1 also be an integer. This difficulty can be resolved by constructing a special index for KEY.1 which provides an order-preserving map of its real-numbered Keys onto positive integers

$\leq M = N^{\lambda}$, for some constant λ. The full-length version of our paper [20] uses the symbol Tb(2) to denote a two-part data structure whose first section is such an index and whose second section is a tree of the type Tc(2) based on the induced integers. Maintaining this index in a dynamic environment is by no means trivial; however, our paper shows it can be done by a combination of the bounded balance methods of [22], the q-fast trie of [19], and the index-methods of Dietz [7], in the context of the tree Tb(2). The resulting complexity $O(\log^{3/2} N)$ has a poor coefficient, but it shows Fredman's formalism for aggregates does not generalize fully to locate-retrievals.

Our main theorem, appearing in [20], regeneralizes the trees Ta, Te, Td, Tf, and Tb for any dimensions $k \geq 2$. This generalization follows from the preceeding results combined with [22]; it states that all the time and space complexities should be multiplied by a factor of $\log^{k-2} N$ when $k \geq 3$.

At least two major open questions remain in multi-dimensional theory. The first is the development of lower bounds for locate-retrievals. The currently known results on lower bounds appear in [11,12,23]. The second question is the generalization of multi-dimensional query theory to polygonal and other nonorthogonal regions. Only three articles have appeared about this subject [10,11,21].

ACKNOWLEDGEMENT

I thank Eric Wolman for suggestions on presentation. This research was supported in part by the Office of Naval Research under Contract N00014-76-C-0914 while the author was at Harvard University.

REFERENCES

1. A. V. Aho, J. E. Hopcroft, and J. D. Ullman, *The Design and Analysis of Computer Algorithms,* Addison-Wesley, Reading, Mass., 1974.

2. Z. Avid and E. Shamir, "A Direct Soultion to Range Search and Related Problems for Product Regions," *Proc. 22nd Annual Symposium on Foundations of Computer Science* (1981), pp. 123-216.

3. J. L. Bentley, "Multidimensional Divide-and-Conquer," *Comm. of ACM 23* (1980), pp. 214-228.

4. J. L. Bentley and H. A. Maurer, "Efficient Worst-Case Data Structures for Range Searching," *Acta Inf. 13* (1980), pp. 155-168.

5. J. L. Bentley and M. I. Shamos, "A Problem in Multivariate Statistics: Algorithm, Data Structure, and Applications," *Proceedings of the Fifteenth Allerton Conference on Communication, Control, and Computing* (1977), pp. 193-201.

6. A. Belour, "Optimal Retrieval Algorithms for Small Region Queries," *SIAM J. on Computing 10* (1981), pp. 721-741.

7. P. F. Dietz, "Maintaining Order in a Linked List," *Proc. of the 14th Annual ACM Symp. on Theory of Computing* (1982), pp. 122-128.

8. D. Dobkin and J. I. Munro, "Efficient Use of the Past," *Proc. of the 21st Annual Symposium on Foundations of Computer Science* (1980), pp. 200-206.

9. H. Edelsbrunner, "Dynamic Data Structures for Orthogonal Intersections Queries," *Technische Universität Graz Report #59* (1980).

10. H. Edelsbrunner, H. A. Maurer, and D. G. Kirkpatrick, "Polygon Intersection Searching," *Inf. Proc. Lett. 14* (1982), pp. 74-79.

11. M. L. Fredman, "The Inherent Complexity of Dynamic Data Structures which Accommodate Range Queries," *Proc. of the 21st Annual Symposium on Foundations of Computer Science* (1980), pp. 191-200.

12. M. F. Fredman, "A Lower Bound on the Complexity of Orthogonal Range Queries," *Journal of ACM 28* (1981), pp. 696-706.

13. D. T. Lee and C. K. Wong, "Quintary Tree: A File Structure for Multidimensional Database Systems," *ACM Transactions on Database Systems 5* (1980), pp. 339-347.

14. D. T. Lee and C. K. Wong, "Finding Intersection of Rectangles by Range Search," *Journal of Algorithms 2* (1981), pp. 337-347.

15. G. S. Lueker, "A Transformation for Adding Range Restriction Capability to Dynamic Data Structures for Decomposable Searching Problems," Technical Report #129, Department of Information and Computer Science, University of California, Irvine, 1979.

16. V. K. Vaishnavi and D. Wood, "Rectilinear Line Segment Intersection, Layered Segment Trees, and Dynamization," *Journal of Algorithms 3* (1982), pp. 160-176.

17. D. E. Willard, *Predicate-Oriented Database Search Algorithms,* Ph.D. thesis, Harvard University, 1978. Also in *Outstanding Dissertations in Computer Science,* Garland Publishing, New York, 1979.

18. D. E. Willard, "New Data Structures for Orthogonal Queries," Technical Report TR-22-78, Center for Research in Computing Technology, Harvard University, 1978. (The earlier draft version of this report.)

19. D. E. Willard, "Two Very Fast Trie Data Structures," to appear in *Proc. 19th Annual Allerton Conference on Communication, Control and Computing* (1981).

20. D. E. Willard, "New Data Structures for Orthogonal Range Queries," Bell Labs Report, 1981; submitted to *SIAM J. Computing.*

21. D. E. Willard, "Polygon Retrieval," *SIAM J. Computing 11* (1982), pp. 149-166.

22. D. E. Willard and G. S. Lueker, "Adding Range Restriction Capability to Dynamic Data Structures," to appear in *Journal of ACM.*

23. A. C. Yao, "On the Complexity of Maintaining Partial Sums," *Proc. of the 14th Annual ACM Symposium on Computing* (1982), pp. 129-145.

SOME PROGRAMMING CONCEPTS FOR CONCURRENT DELETION IN B-TREES

Y. S. Kwong
PHILIPS/MICOM
5250 Ferrier Street
Montreal, P.Q.
Canada H4P 1L4

D. Wood
Dept. of Computer Science
University of Waterloo
Waterloo, Ontario
Canada N2L 3G1

ABSTRACT

In recent years B-trees and their variants have become very popular data structures for storing large amounts of information in secondary storage. Typically such uses of B-trees come hand in hand with concurrent operations on them so that access time can be minimized. For this reason the investigation of "good" algorithms to support concurrency in B-trees is of great interest to both practitioners and theoreticians. Of the three usual operations of SEARCH, INSERT and DELETE, the latter is the most difficult to realize. Therefore we have chosen the problem of concurrent deletion as the major theme of this paper.

Through this problem, we illustrate how several useful programming concepts can be applied together to achieve a new solution. We feel that these programming concepts, together with others in the literature, provide better understanding of the creation and construction of new and better algorithms for supporting concurrency not only in B-trees, but other data structures as well.

Our new solution to the concurrent deletion problem attains a very high degree of concurrency while at the same time requiring only a small constant number of page nodes to be locked at each step of deletion. Moreover, the concurrency control is extremely simple and can be easily realized.

1. INTRODUCTION

In recent years B-trees [BM] and their variants have become very popular data structures for storing large amounts of information in secondary storage, e.g. in implementing dynamic random-access indices for very large databases. For a survey see [C]. Typically such uses of B-trees come hand in hand with concurrent operations on them so that access time can be minimized. For this reason the investigation of "good" algorithms to support concurrency in B-trees is of great interest to both practitioners and theoreticians. The permissible operations are the usual ones for dictionaries, viz., SEARCH, INSERT and DELETE. Dozens of solutions have been proposed in the past few years, including [BS], [E], [KW1], [LY], [MS], [P] and [S]. Examining these solutions carefully, it is straightforward to classify them into two distinct categories (for details, see [KW5]):

Type 1 Solutions: [BS], [E], [KW1], [P], [S]

Updating processes, i.e. INSERT and DELETE, need to lock an unbounded portion of their access path throughout updating and restructuring.

Type 2 Solutions: [E], [KW2], [LY], [MS]

Updating processes only need to lock an, a priori, bounded portion of their access path during updating and restructuring.

It appears that Type 1 solutions impose a lower degree of concurrency than Type 2 solutions. Thus recent interest including ours has focussed attention on Type 2 solutions. Of the four solutions proposed, [E] and [KW2] do not address the problem of deletion whereas [LY] and [MS] do. [MS] while introducing some interesting ideas carry out explicit queue manipulation, thereby blurring the distinction between algorithms and implementations. They also lock O(m) nodes at each restructuring step, where m is the order of the B-tree, which for large m is unacceptable. It is argued in [KK] that a Type 2 solution cannot be expected to perform better than a good Type 1 solution unless the number of nodes locked at each restructuring step is very small, say 3 or 4 and they should not be distributed in more than two levels. [LY] is perhaps the most interesting solution available. They require only a very small number of nodes to be locked at each step, however, private memories are assumed for individual processes.

Because of the numerous solutions proposed, one naturally is led to ask: why another one? Our response is four-fold. First, we do not believe that the solutions available now are the best that can be achieved. Work in [E], [KW2] and [LY] illustrate the recent advances in Type 2 solutions since [MS]. Second, of the three permissible operations on B-trees, DELETE appears to be the most difficult to handle in a concurrent environment and is usually not addressed in most available solutions, including our own [KW2]. Therefore we choose the problem of concurrent deletion as the major theme of this paper. Third, our experience shows that the many programming ideas and concepts that have evolved from different solutions are probably more valuable than the proposed solutions themselves. Through the problem of concurrent deletion, we shall illustrate how several simple programming concepts can be blended together to achieve a reasonably simple solution. Fourth, we believe that concurrent algorithms for B-trees provide excellent examples for the investigation of correctness criteria and proof techniques. While not as simple as the usual examples commonly found in papers on correctness, they are not too complicated and appear to be still manageable. Moreover, attempts to provide correctness definitions and proofs often lead to compromises between the ease of verification and the programming concepts or techniques to be adopted to increase concurrency.

In this paper we shall restrict ourselves to concurrent SEARCH and DELETE to highlight the difficulties involved and illustrate the programming concepts adopted to tackle them. The issue of correctness will be addressed informally. A paper with SEARCH, INSERT and DELETE and a more formal correctness proof is in preparation. The remainder of this paper consists of a further three sections. In Section 2 the B-tree structure and the underlying concurrency control are defined. In Section 3 we address the concurrent deletion problem informally before presenting the detailed solution. In Section 4 we give some concluding remarks.

2. PRELIMINARIES

Definition: A B-tree of order m is a multi-way tree satisfying the following properties:

1) Each node has at most m sons (and is usually represented by a page).
2) Every node, except for the root and the leaves, has at least [m/2] sons.
3) The root has at least two sons.
4) A nonleaf node with j+1 sons contains j routers [KW4] and can be represented as shown in Fig. 1(a), where the P_is are pointers to its sons and the K_is are routers such that all keys in the subtree pointed at by P_{i-1} (P_i) are less than or equal to (greater than) K_i for $1 \leq i \leq j$.
5) All leaves are at the same level. The keys appear in the leaves in ascending order when read from left to right. They are the same as nonleaf nodes except that they have null pointers (actually they should be pointers to actual data). See Fig. 1(b).

Note that we use B-trees as leaf search trees as in [BM] rather than as node search trees, see [K]. Leaf search B-trees are often referred to as B^+-trees and have been used in, for example, IBM's virtual storage access method (VSAM) [IBM]. The removal of a key from an internal node of a node search B-tree is highly non-trival in a Type 2 environment. Leaf search B-trees offer us a simple way out because all keys appear in the leaves, while the internal (non-leaf) nodes are only used to hold routing information or routers.

In order to regulate concurrent accesses we need a concurrency control mechanism. We choose a very simple one with two types of locks that can be imposed on a node by a process, namely p(ioneer)-locks and w(riter)-locks. P-locks (w-locks) are mutually exclusive, i.e., a process cannot p-lock (w-lock) a node which is already p-locked (w-locked) by another process. However, p-locks and w-locks are compatible asymmetrically. A process can w-lock a node which is already p-locked by another process, however, it cannot p-lock a node which is already w-locked. In addition, p-locks and w-locks are fully convertible. A process holding a p-lock (w-lock) can always convert it into a w-lock (p-lock). The compatibility and convertibility relations between the two lock types are illustrated in Fig. 2, where a (directed) solid edge represents (asymmetric) compatibility and a dashed edge represents full convertibility.

Processes can manipulate the locks through three types of indivisible operations. When a process requests a TYPE-lock on a node N, it must execute a lock operation, lock (N, TYPE). If the granting of such a request does not violate the compatibility relation then the process is allowed to continue. Otherwise it is put to sleep in a queue associated with the node N. To release a TYPE-lock on N, a process must execute an unlock operation, unlock (N, TYPE), which relinquishes the lock and wakes up some sleeping processes in the queue according to the compatibility and convertibility relations and some fair scheduling discipline. A process can convert a $TYPE_1$-lock it is holding on the node N into a $TYPE_2$-lock by executing a convert operation, convert (N, $TYPE_1$, $TYPE_2$). If the conversion can be done without violating the compatibility and convertibility relations then the process continues. Otherwise, the operation is undefined.

3. CONCURRENT SEARCH AND DELETE

Our objective is to support an arbitrary number of processes operating concurrently on a leaf search B-tree. These processes are assumed to be asynchronous, each of which is progressing at a finite, but undetermined speed and is performing one of the following dictionary operations:

SEARCH (K, T) : is K in the tree T?
DELETE (K, T) : if K is present in T then remove it.

We assume that the readers are familiar with the original [BM] algorithms which are used as a basis for realizing these two types of processes in a concurrent environment. In the following, we informally describe the major difficulties involved, outline their solutions and highlight the underlying programming concepts which might also be useful for supporting concurrent operations in other data structures.

A SEARCH process is very simple. It makes no changes whatsoever to the data or structure of the tree. Starting at the root, it descends down the tree in the customary manner, examining the routing information at each nonleaf node in order to determine the next node to visit. Before reaching a leaf, it basically behaves as if it is operating in a sequential environment and it is not necessary to lock any nodes before reading. However, once a leaf is reached, the SEARCH process must perform certain operations which are not required in a sequential environment. These operations will be described later in this section after we explain why they are necessary.

A DELETE process is more complicated because it may modify the data as well as the structure of the tree. Usually it has to go through two phases: searching and restructuring. In the searching phase it behaves just like a SEARCH process except that the access path must be "remembered" in case restructuring is necessary. In the restructuring phase the DELETE process retraces its access path. Note that if the access path is not kept then it is necessary to have a father link with each node as in [MS], [KW2]. However maintaining father links is extremely costly since for a B-tree of order m, up to m page nodes have to be modified when a node is removed and this would create an unacceptable amount of secondary storage access. Adopting the access path approach does not resolve all problems. In a Type 2 solution the access path of a DELETE process is not guaranteed to remain fixed during the lifetime of the process, since nodes on it may be removed by a merge operation of some other DELETE process. In this paper we propose that nodes that should be removed after a merge operation are given the status of "ghost nodes." In merging a node N with a brother node B, a new node N_n is created into which the contents of B and N are copied. After linking up N and

B to the new node, both N and B are marked as ghosts. See Fig. 3. Thus on retracing the access path, each node is checked to see if it is a ghost node in which case the process moves onto the merged node. Moreover since this in turn may be a ghost node it is also linked to its merged version and so on. Eventually, of course, ghost nodes must be removed, this is done by way of time-stamping all ghost nodes and processes on their creation, as is done in [KL] for example. Then whenever a ghost node is older than all processes currently in the system, it may be removed without affecting the integrity of B-tree. Note that this notion of using ghost nodes is very similar to the concept of side-branching adopted in the Type 1 solution in [KW1]. Both are variations of the concept of using extra copies of information to increase concurrency and maintain integrity.

A similar problem is encountered when a process retraces a node on its access path which was involved in a rotation operation by another DELETE process. The key and pointer it is trying to remove may no longer be there because of the rotation. Let us assume, for simplicity, that a key-pointer pair (K, P) is to be rotated into a node N from its right brother B. See Fig. 4(a). The case when B is to the left of N is analogous. Instead of actually altering the contents of N and B, the key rotation is achieved by creating two new nodes N_n and B_n into which N and B are copied and the key and pointer are removed from B_n and added to N_n. The separating key in their father node also needs to be modified. Hence a restructuring process searching B for a key less than or equal to K should be routed to N_n. Otherwise, it should be routed to B_n. A restructuring process searching N for any key should be routed to N_n. After establishing these two links, both N and B can then be marked as ghosts. See Fig. 4(b). Thus a node needs to have two links for routing purposes. When only one pointer needs to be maintained, we could set both links to point at the same node or use one link and leave the other undefined.

Adopting this approach, a node in our data structure can be either a ghost node or a non-ghost node. We shall refer to non-ghost nodes as being active. There are two types of ghost nodes: merged ghosts and rotatory ghosts, depending on whether they are created as a result of node merging or key rotation. A rotatory ghost from which a key-pointer pair is supposed to be removed is referred to as a left (right) rotatory ghost if the key and pointer are rotated into a node to its left (right). In previous paragraphs, we omitted discussion of what a SEARCH process should do on reaching a leaf. When it is at the leaf level, the leaf it is examining may be a ghost. Hence ghost chasing (skipping over all ghost nodes) is necessary until an active node is found. Then it can check if the key K is actually present and report accordingly. To ensure that it reports correctly (at least at that particular instance), it is necessary to have a node locked before checking for K. In our solution, w(rite)-locks are being used by a SEARCH process. Of course, all locks have to be released before the process terminates.

For a DELETE process ghost chasing is also necessary during restructuring when it tries to lock a node in retracing its access path. To ensure consistency restructuring DELETE processes sharing the same access path should not overtake one another. This can be achieved by requiring a DELETE process to hold onto a node until it can lock its father node successfully. This concept of lock-coupling [KW1] is well-known and has been used in one form or another in almost all solutions in the literature. Three nodes have to be locked at each restructing step: the node from which a key-pointer pair is to be removed, its father node and an immediate brother. However, if only w-locks are being used, a deadlock situation may occur when two processes are racing toward a common ancestor on their access paths. As shown in Fig. 5(a), process P_1 has locked node N and its father F and is trying to lock its immediate brother B which is being locked by another process P_2. In our solution such potential deadlocks are avoided by adopting the concept of asymmetric locking proposed in [MS] and requiring a DELETE process to use another lock type, p(ioneer)-locks, whenever it wants to move to the next restructuring step. A process with a p-lock on a node should not change its content; it simply uses a p-lock to announce its intention of using that node. A p-locked node serves as a buffer zone between two competing processes racing towards a common ancestor. To ensure deadlock-freedom, p-locks and w-locks are asymmetric: a process can w-lock a node p-locked by another process, but not the other way round. See Fig. 5(b).

The details of our solution are presented in the following algorithms for SEARCH and DELETE, using a pidgin-Pascal language.

```
function SEARCH (K, T) : boolean ;
/*this function searches for the key K in tree T and returns true or false accordingly*/
    node   :=  root of T;
    /*the boolean function FIND is called upon to do the actual searching*/
    SEARCH :=  FIND (K, node, 'search', dummy);
    unlock (node, w);

function FIND (KEY, NODE, PROCESS, ACCESS_PATH) : boolean ;
/*this boolean function searches for the key KEY in the subtree rooted at NODE; if the calling
  process is a deleter then the access path will be returned*/
    ACCESS_PATH := nil ;
    /*decend down the tree; no locking at levels above the leaves*/
    while  (NODE is not a leaf)  do
        if  PROCESS = 'delete'  then  add NODE to ACCESS_PATH  endif ;
        determine appropriate son of NODE;
        NODE := son
    endwhile ;
    /*lock an appropriate active leaf and determine if KEY is there*/
    lock (NODE, w);
    SKIP_GHOSTS_LOCK_ACTIVE (NODE, w, KEY);
    FIND := KEY is in NODE;

procedure  SKIP_GHOSTS_LOCK_ACTIVE (NODE, TYPE, KEY);
/*this procedure assumes that NODE is already locked; it skips over all ghost nodes until an
  active node linked to NODE is eventually locked*/
    while  (NODE is not active)  do   skip over all ghost nodes
        old := NODE;
        if    NODE is a right-rotatory ghost & KEY > rightmost key in NODE
           or NODE is a left-rotatory ghost & KEY <= leftmost key in NODE
        then  NODE := right-link (NODE)  else  NODE := left-link (NODE)  endif ;
        unlock (old, TYPE);
        lock (NODE, TYPE);
    endwhile ;

procedure  DELETE (K, T);
/*this procedure removes the key K from the tree T if it is present; it consists of two
  phases: searching and restructuring*/
    /*searching for K in the tree T*/
    current   :=  root of T;
    if  not  FIND (K, current, 'delete', access_path)  then
        /*K is not in T; simply terminate after unlocking*/
        report K not found;
        unlock (current, w)
    else
        /*K is in the tree and is in current; prepare for removal and restructuring*/
        convert (current, w, p);
        lock (father_of_current, p);
        SKIP_GHOSTS_LOCK_ACTIVE (father_of_current, p, K);
        convert (current, p, w);
        SKIP_GHOSTS_LOCK_ACTIVE (current, w, K);
        $k_r$ := K;  $p_r$ :=  nil ;
        /*if current is not safe then restructuring is necessary; this usually consists of a
          number of merging and then possibly by a rotation*/
        while  (current is not d-safe)  do
            convert (father_of_current, p, w);
            SKIP_GHOSTS_LOCK_ACTIVE (father_of_current, p, $k_r$);
            w-lock immediate bothers of current;
            if  both immediate brothers are minimal  then
                /*merging operation*/
                let imb be an immediate minimal brother and w-unlock the other immediate
                    brother if it exists;
                MERGE (current, father_of_current, imb, $k_r$, $p_r$);
                unlock (imb, w);
                unlock (current);
                current := father_of_current;
                convert (current, w, p);
                lock (father_of_current, p)
                SKIP_GHOSTS_LOCK_ACTIVE (father_of_current, p, $k_r$);
                convert (current, p, w);
                SKIP_GHOSTS_LOCK_ACTIVE (current, w, $k_r$)
            else
                /*rotation operation*/
                let isb be an immediate d-safe brother and w-unlock the other immediate
                    brother if it exists;
                if  current is to the left of isb  then  dir := 'left'
                else  dir := "right'  endif ;
                ROTATE (current, father_of_current, isb, dir, new);
                lock (new, w);
                unlock (current, w);
                current := new;
                convert (father_of_current, w, p)
            endif
        endwhile ;
        /*actual removal of key and pointer*/
        convert (father_of_current, p, w);
        SKIP_GHOSTS_LOCK_ACTIVE (father_of_current, w, $k_r$);
        REMOVE (current, father_of_current, $k_r$, $p_r$);
        unlock (current, w);
        unlock (father_of_current, w)
    endif ;

procedure  MERGE (NODE, FATHER, BROTHER, KEY, PTR);
/*this procedure merges NODE and BROTHER by creating a new node into which keys and pointers
  are copied; NODE and BROTHER are then marked as ghost nodes*/
    let $k_s$ be the separating key in FATHER for NODE and BROTHER;
    CREATE (new_node);
    copy all keys and pointers in NODE (except KEY and PTR) and BROTHER into new_node
      together with $k_s$;
    left_link (BROTHER) := new_node;
    left_link (NODE)    := new_node;
    set the pointer to NODE in FATHER to new_node;
    set the pointer to BROTHER in FATHER to new_node;
    KEY  :=  $K_s$;  PTR  :=  pointer to NODE in FATHER;
    make NODE and BROTHER merged ghosts;

procedure  ROTATE (NODE, FATHER, BROTHER, DIR, NEW);
/*this procedure rotates a key and a pointer from BROTHER into NODE through their FATHER*/
    let $k_s$ be the key in FATHER separating NODE and BROTHER;
    let $p_l$, $k_l$ ($p_r$, $k_r$) be the leftmost (rightmost) key and pointer in BROTHER;
    if DIR = 'left'  then  ptr := $p_l$; k := $k_l$
    else  ptr := $p_r$; k := $k_r$  endif ;
    CREATE (new_node);
    copy all keys and pointers from NODE to new_node together with $k_s$ and ptr;
    left_link (NODE) := new_node;
    set the pointer to NODE in FATHER to new_node;
    make NODE a ghost;
    replace $k_s$ in FATHER by k;
    CREATE (new_brother);
    copy all keys and pointers from BROTHER (except ptr and k) into new_brother;
    left_link (BROTHER) := new_brother;
    right_link (BROTHER) := new_node;
    replace the pointer to BROTHER in FATHER by pointer to new_brother;
    if  DIR = 'left'  then  make BROTHER a left-rotatory ghost
    else  make BROTHER a right-rotatory ghost  endif ;
    NEW  := new_node;

procedure  REMOVE (NODE, FATHER, KEY, PTR);
    CREATE (new_node);
    copy all keys and pointers from NODE (except KEY and PTR) into new_node;
    left_link (NODE)  :=  new_node;
    set the pointer to NODE in FATHER to new_node;
    make NODE a ghost;
```

4. CONCLUDING REMARKS

Let us first summarize the contribution of this paper. We have obtained a new solution to the concurrent deletion problem in B-trees. The solution is reasonably simple and straightforward. Through the investigation of this problem, we have identified a number of difficulties which are very common when supporting concurrency in tree structures. We have illustrated how some simple programming concepts can be applied to resolve these difficulties. These concepts include the use of ghost nodes, lock-coupling, and asymmetric locking. Some of them are our own; some have been used in the literature. We feel that the accumulation of this wealth of concepts and techniques will shed more light on how to construct new and better concurrent algorithms for B-trees and other data structures.

The interesting features of our solution include:

- Father links are not used - it appears that Type 2 solutions using father links (e.g. [MS], [KW2]) have little if any chance of performing better than a good Type 1 solution in practice.
- A SEARCH process does not have to lock a node before reading except at a leaf. Lock-coupling is not required during searching, making a SEARCH process very similar to the original [BM] algorithm.
- Since lock-coupling is not employed by SEARCH processes, it is impossible to drive away processes as in [KW2] and many Type 1 solutions. However, queue processing as in [MS] is avoided by using ghost nodes; a process either exits naturally from a node or is re-routed via ghost nodes.
- Only two levels of nodes need to be locked at each restructuring step and it suffices to lock at most three nodes at any time. Although we lock both immediate brothers of a node at the same time in our algorithm for the sake of simplicity, this is, in fact, unnecessary.
- A SEARCH process may see a ghost on its way down the tree, however, it suffices to do ghost chasing only at the leaf level.
- It is impossible for deadlocks or livelocks to occur [Kw].

Some unpleasant features of the solution include:

- The number of ghosts linked together is only bounded by the number of nodes in the original B-tree, and if INSERT is included then ghost linking can be of unbounded extent.
- There is an additional problem of garbage collection introduced because of ghost nodes, although time-stamping can be realized fairly easily.

A surprising feature of our solution is that livelock cannot occur. However with a little thought this is seen to be an immediate implication of only providing for SEARCH and DELETE. If we add INSERT to this solution then livelock may occur. This follows from the fact that the INSERT operation can cause nodes to split when they are overfilled. Hence a SEARCH process may always be in the state of just about to find its search key when the node it is looking at is split into two. This is however a highly unlikely state of affairs.

The extent of ghost linking might be reduced by using a path compression technique, namely each time a sequence of ghost nodes is scanned their linkage is compressed so that each of them is directly linked to the "real" successor node on the path. However, such a technique may render a correctness proof more difficult.

ACKNOWLEDGMENT

Work reported herein was partially supported by the Natural Sciences and Engineering Research Council of Canada under Grant Nos. A-3042 and A-7700.

REFERENCES

[BM] Bayer, R. and McCreight, E. Organization and maintenance of large ordered indexes. Acta Informatica 1 (1972), 173-189.

[BS] Bayer R. and Schkolnick, M. Concurrent operations on B-trees. Acta Informatica 9 (1977), 1-21.

[C] Comer, D. The ubiquitous B-tree. ACM Computing Surveys 11 (1979), 121-138.

[E] Ellis, C. S. Concurrent search and insertion in 2-3 trees. Acta Informatica 14, 1 (1980), 63-86.

[GS] Guibas, L. J. and Sedgewick, R. A dichromatic framework for balanced trees. Proc. 19th Annual Symposium on Foundations of Computer Science (1978), 8-21.

[IBM] OS/VS Virtual Storage Access Method (VSAM) Planning Guide. Order No. GC26-3799, IBM, Armonk, N.Y.

[K] Knuth, D. E. The Art of Computer Programming, Volume 3: Sorting and Searching. Addison-Wesley, Reading, MA (1973).

[KK] Kriegel, H. P. and Kwong, Y. S. Insertion-safeness in balanced trees. Computer Science Technical Report 81-CS-10, McMaster University.

[KL] Kung, H. T. and Lehman, P. L. A concurrent database manipulation problem: binary search trees. ACM Trans. on Database Systems 5, 3 (1980), 339-353.

[Kw] Kwong, Y. S. On livelocks in parallel programs. Intl. Journal of Computer Mathematics, Part 1: 10 (1981), 121-135; Part 2: 10 (1982), 201-212.

[KW1] Kwong, Y. S. and Wood, D. A new method for concurrency in B-trees. IEEE Trans. on Software Engineering SE-8, 3 (1982), 211-222.

[KW2] Kwong, Y. S. and Wood, D. Concurrency in B-trees, S-trees and T-trees. Computer Science Technical Report 79-CS-17, McMaster University.

[KW3] Kwong, Y. S. and Wood, D. Concurrent operations in large ordered indices. Proc. 4th Intl. Symposium on Programming. Springer-Verlag Lecture Notes in Computer Science 83 (1980), 207-222.

[KW4] Kwong, Y. S. and Wood, D. On B-trees: routing schemes and concurrency. Proc. ACM/SIGMOD Intl. Conf. on Management of Data (1980).

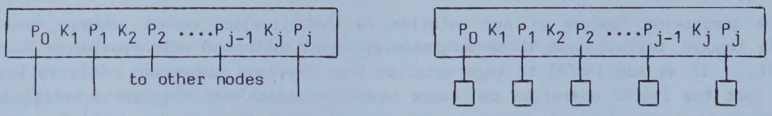
[KW5] Kwong, Y. S. and Wood, D. Approaches to concurrency in B-trees. Proc. MFCS '80. Springer-Verlag Lecture Notes in Computer Science 88 (1980), 402-413.

[LY] Lehman, P. L. and Yao, S. B. Efficient locking for concurrent operations on B-trees. ACM Trans. on Database Systems 6, 4 (1981), 650-670.

[MS] Miller, R. E. and Snyder, L. Multiple access to B-trees. Proc. John Hopkin Conf. on Information Sciences and Systems (1980).

[P] Parr, J. R. An access method for concurrently sharing a B-tree based indexed sequential file. Computer Science Report 36, University of Western Ontario (1977).

[S] Samadi, B. B-trees in a system with multiple users. IPL 5, 4 (1976), 107-112.


P_0 K_1 P_1 K_2 P_2 ....P_{j-1} K_j P_j
to other nodes
P_0 K_1 P_1 K_2 P_2 ....P_{j-1} K_j P_j

(a) A non-leaf node (b) A leaf node

Fig. 1 Nodes in a B-tree.

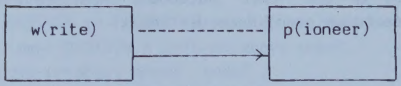
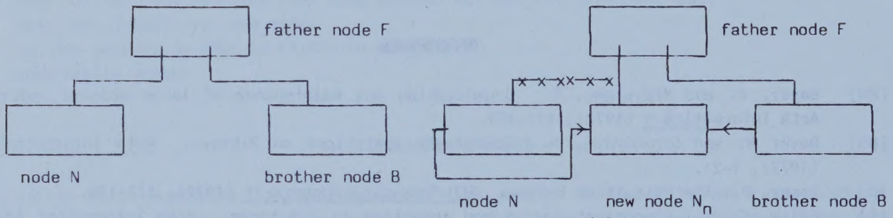
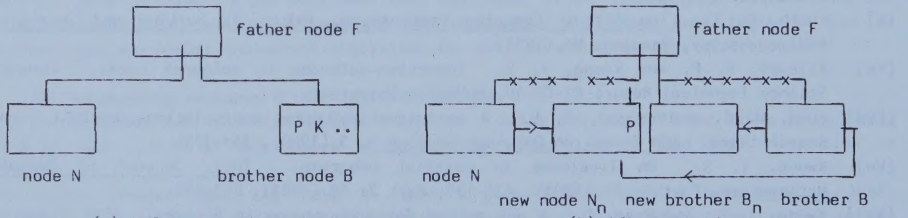

w(rite)
p(ioneer)

Fig. 2 Compatibility and convertibility relations.


father node F
node N
brother node B
father node F
node N
new node N_n
brother node B

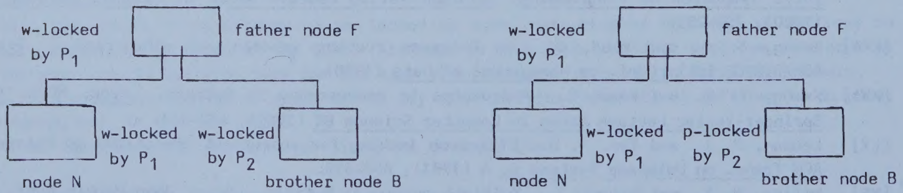
(a) Before merging (b) After merging

Fig. 3 Node merging.


father node F
P K ..
node N
brother node B
father node F
P
node N
new node N_n
new brother B_n
brother B

(a) Before rotation (b) After rotation

Fig. 4 Key rotation.


w-locked by P_1
father node F
w-locked by P_1
w-locked by P_2
node N
brother node B
w-locked by P_1
father node F
w-locked by P_1
p-locked by P_2
node N
brother node B

(a) P_1 tries to w-lock node B
P_2 tries to w-lock node F

(b) P_1 tries to w-lock node B
P_2 tries to p-lock node F

Fig. 5 Deadlocks and their avoidance.

BINARY SEARCH TREES WITH LOCAL ROTATION

SHOU-HSUAN STEPHEN HUANG
Department of Computer Science
University of Houston
Houston, Texas 77004

C.K. WONG
IBM Thomas J. Watson Research Center
P.O. Box 218
Yorktown Heights, New York 10598

ABSTRACT: A parameterized binary search tree called mR tree is defined in this paper. A user is allowed to select a level of balance he or she desires. SR tree is a special case of mR tree when m=1. There are two concepts in SR trees: (1) local balancing scheme that balances the tree locally; (2) consecutive storage for brother nodes that reduces pointer space. Although we may introduce empty nodes into the tree, we can show that only 1/8 of the nodes may be empty on the average. So it may still be advantageous in cases when record sizes are small. Insertion (and deletion) into SR trees can be done in time $h + O(1)$ where h is the height of the tree. The average searching time for SR trees is shown to be $(12/7)\ell n\, n$ or $1.188 \log_2 n$ where n is the number of keys. Generalization of the results of SR trees to mR trees is also given.

I. INTRODUCTION

Binary search trees have been studied for quite a long time [1,2]. On average, it takes O(log n) time to find a record in the tree where n is the number of keys in the tree. However, the worst case cost can be as large as O(n). There are several ways to balance the trees. However, most of the schemes involve global properties that are difficult to maintain. We present one way to balance the tree locally, called SR (Single Rotation) trees.

Another problem SR trees address is the storage efficiency of binary trees. If the record size is small, then it may not be wise to spend so much storage space for pointers. For example, if the size of the record is the same as that of a pointer, we actually use only one third of the storage for records. We can reduce one pointer per node if the two sons of a node appear in consecutive memory locations. Notice that in doing so, we may introduce some empty nodes into the tree. So it may not be advantageous for all cases.

Such a tree structure with a local balancing method is indeed used in a name table management scheme for a new APL interpreter system with very good experiment results for average search time and storage efficiency [3].

We first study analytically the search time and the storage efficiency of SR trees in Section 2. Insertion and deletion algorithms are also given. Then we generalize it to the class of DR (Double Rotation) and mR trees in Section 3 and study their behavior as well. The mR trees defined here is equivalent to the k-locally balanced trees of Walker and Wood [7]. However they fail to show the relation between the average access cost and the degree of balance.

II. BINARY SR TREES

Definition 1.

A Binary SR tree is a binary tree that satisfies the following conditions:

(1) A node (except for a leaf) has 2 sons, one of them may be empty, but not both.

(2) A node N cannot have sons if N's brother is empty.

An example of an SR tree is given in Figure 1 below. Since some of the nodes may be empty, we have

Lemma 1: The storage utilization of an SR tree is at least $3n/(4n + 1)$ where n is the number of keys in the tree.

INSERTION

The algorithm to insert a new key into an SR tree is almost the same as insertion into an ordinary tree execpt that we may have to do some balancing at the end. The cost of insertion (and deletion) is measured in terms of number of nodes visited (and possibly modified) in order to perform the operation. In this case, the insertion costs at most h + 1 node visits.

DELETION

Deletion is more complicated than insertion as in most cases. Suppose we want to delete a key from node N (with father F and brother B), depending on the number of sons, we divide the algorithm into the following cases.

Case 1: Node N has no sons.

If nodes N and B do not have sons, deletion is easy. If B has sons, then we have to delete both nodes N and F, replace F by B and then insert the key in F into the subtree rooted at B.

Case 2: Node N has one son.

Replace node N with its non-empty son.

Case 3: Node N has 2 sons.

Find a node W such that it contains the next lower (or higher) key in the tree. Node W must be in the subtrees of N, otherwise N cannot have 2 sons. Replace the key (x) in N by the key (x) in W. The problem now becomes how to delete x' in W. It is easy to see that node W cannot have 2 sons, thus it must have one or no son. So we can delete x' in W by either case 1 or case 2 above.

The cost to delete a key is h + O(1). The reason is that we have to go from the root to leave at most once (stop at halfway to do some modification in case 3).

AVERAGE STORAGE UTILIZATION

To compute the average storage utilization of binary SR trees, we perform a two level analysis as in Yao [4]. Since empty nodes can only appear at the frontier, we are interested in the distribution of these nodes.

There are 3 types of subtrees in the frontier. Figure 1 shows an example of SR tree decomposed into these types. According to [1,pp. 679-680], the average number of type i subtrees are

$$A_1(n) \sim \frac{4}{35}(n + 1), \; A_2(n) \sim \frac{5}{35}(n + 1), \; A_3(n) \sim \frac{3}{35}(n + 1).$$

Among all nodes in the tree, only one node per type 2 subtree is empty, all the rest are full. So we have about $n + 5/35(n + 1)$ nodes in an SR tree. By the definition of the storage utilization, we have $u \sim \dfrac{n}{n+(5/35)(n+1)} \sim 7/8.$

Although we wasted 1/8 of the nodes on average, we save one pointer per node. Thus in cases when records are small, we may actually save storage by using SR trees.

SEARCH TIME

In searching the tree, we have to reach the left son first to reach the right son. Since their memory locations are adjacent, we neglect the time to move from the left son to the right son and assume that reaching the left or right son takes the same amount of time. Thus we can measure the search time by the path length of the key to be searched as in the case of ordinary search trees.

It is easy to see that in the worst case we have to search n/2 nodes to get The average cost is given in the next theorem.

THEOREM I:

The average number of comparisons in a successful search in SR trees of n nodes is approximately $\frac{12}{7}$ (ℓn n) + O(1).

To find the average search time, we need the following definitions.

$I(T_n)$: internal path length of a tree of n keys,

$E(T_n)$: external path length of a tree with n keys,

$C_n (C'_n)$: average number of comparisons in an (un-)successful search assuming that each of the n+1 intervals between keys is equally likely.

The definitions above and their relations below are taken from [1,p.410] and [6].

$$C_n = 1 + \frac{I(T_n)}{n} \qquad (1)$$

$$C'_n = \frac{E(T_n)}{n + 1} \qquad (2)$$

$$C_n = (1 + \frac{1}{n})C'_n - 1 \qquad (3)$$

From (1) and (3), we get

$$I(T_n) = (C_n - 1)n \qquad (4)$$

and

$$C'_n = \frac{(C_n + 1)}{1 + 1/n} \qquad (5)$$

Lemma 2: On average, there is 2/7 probability that a single rotation will occur when inserting a new key into an SR tree of a large number of keys.

It is easy to see that inserting a key into a binary tree of n-1 keys will increase the internal path length by C'_{n-1} on average. In the case of SR trees, we have

$$I(T_n)=I(T_{n-1}) + \frac{2}{7}(C'_{n-1} -1) + \frac{5}{7}(C'_{n-1}) \qquad (6)$$

The insertion algorithm can easily be generalized to DR trees. The only difference is at the end, we have to check a subtree of up to 7 nodes to see if the tree is locally balanced. ("Double Rotation" is used to describe this balancing operation. It is slightly more general than its traditional definition [6,p. 240] which involves 3 nodes.) Since 3 of those nodes are included in a path of the tree, we have

Lemma 4: The insertion algorithm for DR trees requires at most $h + 4$ node visits.

The deletion alogrithm is basically the same as SR trees. The only difference is when deleting a node without sons, we have to do more checking. So the worst case cost of a deletion is that of insertion plus a constant. A simple counting gives us:

Lemma 5: The deletion algorithm for DR trees requires at most $h + 7$ node visits.

To compute the average storage utilization, we have to divide the subtrees into six types and do a three level analysis. The average numbers of subtrees are approximately:

$$A_1 \sim 56/1599,\ A_2 \sim 84/1599,\ A_3 \sim 36/1599,$$
$$A_4 \sim 24/1599,\ A_5 \sim 45/1599,\ A_6 \sim 35/1599.$$

From the above distribution, we conclude that the average storage utilization of DR trees under random insertion is about 0.9083.

We can also compute the average access cost of DR trees by using the same method. The result is listed below together with the result of SR and binary trees for comparison.

	Average Access Cost	Worst Case Insertion Cost
binary trees	$2 \ln n$	height
SR trees	$1.714 \ln n$	height + 1
DR trees	$1.576 \ln n$	height + 4

(Note that the heights of SR and DR trees are smaller than ordinary binary trees).

As we can see, we can decrease the access cost by adding stronger balancing conditions to the tree. Note that the height of the tree also decreases as we go from binary trees to DR trees, so the costs for DR insertion may be smaller for large n.

There are several possible ways that we can generalize our results further. First, we can put more nodes in consecutive locations, thus reducing the pointer space further. For example, in DR trees, we can store the four nodes of the same grandfather together. However, this may introduce more empty nodes. In fact, the average storage utilization will drop from 90.83% to 79.32% according to the distribution we derived in the last section. We will run into the same problem as in the first case, i.e., the storage will be very poor if we put all sons of a node in consecutive memory locations. Another way to generalize the idea is to increase the level of local balancing. That is, follow the same direction of SR to DR.

since each rotation reduces the search time by 1. Using (5) to replace C'_{n-1} by C_{n-1}, we get

$$I(T_n) = I(T_{n-1}) + C'_{n-1} - \frac{2}{7}$$

$$= I(T_{n-1}) + \frac{n-1}{n}(C_{n-1} + 1) - \frac{2}{7}$$

i.e.,

$$(C_n - 1)n = (C_{n-1} - 1)(n - 1) + \frac{n-1}{n}(C_{n-1} + 1) - \frac{2}{7} \qquad (7)$$

by (4). So we have the following recurrence relation

$$nC_n = (n - \frac{1}{n})C_{n-1} - \frac{1}{n} + \frac{12}{7} . \qquad (8)$$

From (7), we can get an approximation of (8) by replacing (n-1)/n by 1. Notice that this will only increase the cost on C_n. So,

$$nC_n \sim nC_{n-1} + \frac{12}{7}$$

i.e.,

$$C_n \sim C_{n-1} + \frac{12}{7n} \qquad (9)$$

Solving (9), we get

$$C_n \sim \frac{12}{7} H_n + \alpha$$

where α is a constant and

$$H_n = \sum_{i=1}^{n} \frac{1}{i} .$$

By using the boundary condition $C_1=1$, we get

$$C_n \frac{12}{7}(H_n - 1) + 1 \qquad (10)$$

Since

$$H_n = \ell n\, n + r + O(1/n)$$

where r=0.57721...(see [5,p.47]), we have

$$C_n \sim \frac{12}{7} \ell n\, n + O(1)$$

The corresponding value for ordinary binary trees is 2 ℓn n. So the average search time for SR trees is about 14.3% less than that of ordinary binary trees.

III. GENERALIZATION

We can generalize SR trees to DR trees in the following manner.

DEFINITION 2:

Binary DR trees are those SR trees that also satisfy the following condition: A node cannot have great grandsons if any of its grandsons are empty.

Lemma 3: The worst case storage utilization of DR trees is the same as SR trees. The worst height of DR trees is $\lceil (n+5)/4 \rceil$.

From the above Lemma, we know that we do not improve the worst storage efficiency going from SR to DR trees. However, we will show that the average storage utilization will be increased. The worst height of DR trees is about half of the size of the SR trees, a significant improvement.

To define a class of trees that generalize the SR trees, we need the following terms. We will call sons, grandsons, ..., of a node the 1 level, 2 level, ..., descendants. Also, we define 1R trees to be SR trees.

DEFINITION 3: An mR tree is an (m-1)R tree such that a node cannot have m+1 level descendants if any of its m level descendants is empty.

If we ignore empty nodes, ordinary binary trees can be considered 0R trees. The insertion and deletion algorithms can be easily generalized from the previous algorithms for SR and DR trees. The worst case storage utilization is always about 3/4 as in the cases of SR and DR trees. However, the height of the mR tree decreases as m increases.

Lemma 6: The height of an mR tree with n keys is at most

$$\left\lceil \frac{n - 2^m + 1}{2^m} + m \right\rceil .$$

When m increases, the mR tree looks more and more like a completely balanced tree. As a matter of fact, when

$$m = \lceil \log_2 (n + 1) - 1 \rceil$$

mR tree becomes completely balanced. Note that in this case, the upper bound of the height in Lemma 6 reduces to $\lceil \log_2 (n+1) \rceil$, the optimal value. The average storage utilization will improve somewhat but not much. The access cost will drop to ℓn n. Of course, the insertion and deletion cost will increase as m increases. The insertion cost is basically the height plus the cost of the balancing operation.

Lemma 7: The worst case cost of inserting a key into an mR tree is $h + 2^{m+1} - (m + 2)$ node visits.

Note that although the cost to balance the tree increases, the height decreases. A user may select the best m for his or her own application.

We shall study C_n^m, the average number of comparisons for a successful search of mR trees of n keys next. The saving of C_n^m is due to the local rotations that were performed during insertion. Thus we need to find out the probability of local rotations per insertion.

An mR subtree of height m+1 must have $2^m - 1$ nodes on the first m levels and up to 2^m nodes on the lowest level. A *typi i subtree* is such a subtree with i nodes (hence $i-2^m + 1$ nodes on the lowest level), i = p-1,..., 2p-1 where $p = 2^m$. To simplify our analysis, we include type p-1 subtree of height m.

Following Yao's analysis [4,p.170], we can find an asymptotic distribution u_i of these subtrees on the frontier of an mR tree.

$$u_{p-1} = \frac{1}{p+1}\,\frac{1}{2p+1}\,(H(2p)-H(p))^{-1},$$

$$u_i = \frac{1}{i+1}\,\frac{1}{i+2}\,(H(2p) - H(p))^{-1}, \quad p \leq i \leq 2p - 1.$$

Therefore,

$$\sum_{i=p}^{2p-1} u_i = \frac{1}{2p+1}\,(H(2p) - H(p))^{-1}.$$

Now our analysis departs from Yao's. For each type i subtree, there are positions that will cause rotation to occur. Thus the average possibility for a rotation is

$$\alpha_m = \sum_{i=p}^{2p-2} 2(i-p+1)u_i$$

$$= 2-(H(2p)-H(p))^{-1}$$

Note that $\alpha_0 = 0, \alpha_1 = 2/7$ and $\alpha_2 = 678/1599$; all agree with our previous analysis. It is well known [1, p. 74] that $H(p) \simeq \ell n\ p + 0.58 + 1/2p + \ldots$. Thus

$$\alpha_m \simeq 2-\frac{1}{\ell n 2 - 2^{-(m+2)}}$$

Following the same analysis as in the case of m=1, we can compute C_k^m as follows.

$$nC_n^m = \left(n-\frac{1}{n}\right)C_{n-1}^m - \frac{1}{n} + 2-\alpha_m$$

Thus

$$C_n^m \simeq C_{n-1}^m + (2-\alpha_m)$$

Solving the above recurrence relation, we have

$$C_n^m \simeq (2-\alpha_m)H_n + \alpha$$

where α is a constant. With the boundary condition $C_1=1$, we have $\alpha=\alpha_m-1$

So,

$$C_n^m \simeq (2-\alpha_m)(H_n-1)+1$$

$$\simeq \frac{\ell n\ n}{\ell n\ 2-2^{-(m+2)}} + O(1)$$

$$\rightarrow \log_2 n \text{ as } m \rightarrow \infty$$

Note that the insertion and deletion costs are increasing functions of m. Thus we should limit m to a small number to take advantage of local balance. Since α_m does not change much for m greater than 3, the saving of C_k^m is limited. Thus m=2 or 3 are probably the best choice. For m=2, the average access cost is $1.576\ \ell n\ n + O(1)$, compared to $1.443\ \ell n\ n + O(1)$ for AVL trees. The difference reduces to 4.57% for m=3.

IV. CONCLUSION

We defined a class of binary trees with a parameter m that specifies the level of balance. Higher m gives us a more balanced tree (smaller height) at the expense of higher insertion and deletion costs. At one extreme (m=0), mR tree is simply binary tree (ignoring the empty nodes); at the other extreme, mR tree is the completely balanced tree. Binary SR tree is studied in detail in Section 2. Since AVL trees also use rotations to babalance the tree, it may be worthwhile to compare these two classes of trees. Although AVL trees have a smaller search time, SR trees have the following advantages: (1) although both trees have fairly efficient insertion algorithms, the deletion algorithm for AVL trees may require up to O(h) rotations whereas SR trees require only 1. Thus the worst-case deletion cost is 3h+O(1) (vs. h + O(1)). (2) For the same reason as (1), SR trees provide a better environment for concurrent accessing. For example, locking is easier to handle in SR tree because of the local balancing scheme. Furthermore, if the tree is distributed over several locations, a SR tree is certainly more attractive than an AVL tree because of the non-backtracking deletion algorithm. (3) For records of small size, SR trees use less storage than AVL trees.

From the above discussion, we can see that SR trees are best suited for records of small size in the main memory where storage efficiency is of utmost concern. We also derived the average cost for mR trees in general. The result agrees with our analysis for m=1 in Section 2 and the analysis of Guibas and Sedgewick [8, p. 18]. Note that Guibas and Sedgewick derived C_n^m for m=1 only. From C_n^m we conclude that the average access cost of mR tree is fairly close to that of AVL trees even for very small m (2 or 3). However 2R or 3R trees are much easier to maintain (under insertions and deletions) than AVL trees.

REFERENCES

[1] D.E. Knuth, The Art of Computer Programming, Vol.3, Addison-Wesley, Reading, Mass. 1973.

[2] Th. Ottmann, H.W. Six and D. Wood, The Analysis of Search Trees: A Survey, Lecture Notes in Computer Science 100, pp. 234-249,1981.

[3] W.M. Ching, Private Communication.

[4] A.C. Yao, On Random 2-3 Trees, Acta Informatica, 9 (1978), 159-170.

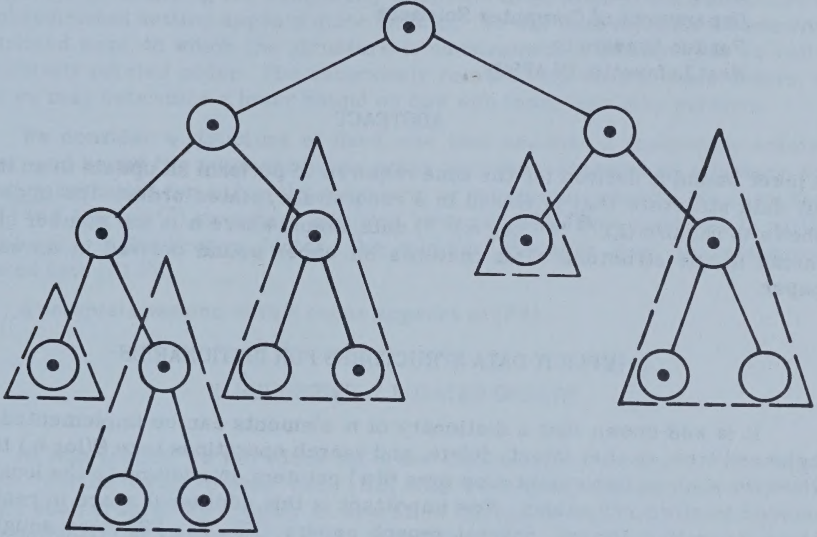
[5] D.E. Knuth, The Art of Computer Programming, Vol.1, Addison-Wesley, Reading, Mass. 1968.

[6] E.M. Reingold, J. Nievergelt and N. Deo, Combinatorial Algorithms: Theory and Practice, Prentice-Hall, Englewood, New Jersey, 1977.

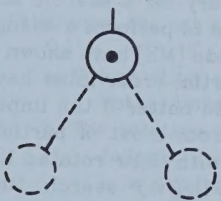
[7] A. Walker and D. Wood, Locally Balanced Binary Trees, The Computer Journal, Vol. 19, No. 4, pp. 322-325, 1976.

[8] L.J. Guibas and R. Sedgewick, A Dichromatic Framework for Balanced Trees, 19th Symposium on Foundations of Computer Sciences, 1978, pp.8-21.

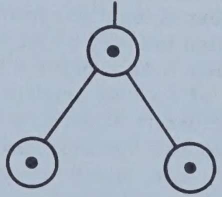
Figure 1: An Example of SR Trees with 13 Keys



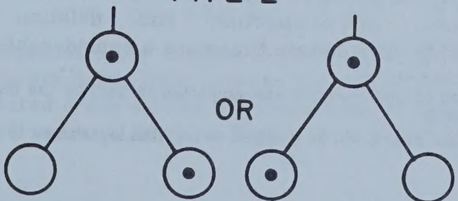
TYPE 1



TYPE 3



TYPE 2



RECURSIVELY ROTATED ORDERS AND IMPLICIT DATA STRUCTURES: A LOWER BOUND* (Extended Abstract)

GREG N. FREDERICKSON
Department of Computer Sciences
Purdue University
West Lafayette, IN 47907

ABSTRACT

A lower bound is derived for the time required to perform an update in an implicit data structure that is stored in a recursively rotated order. The update is shown to require $\Omega(2^{\sqrt{2 \log n}} (\log n)^{1/2})$ data swaps, where n is the number of elements in the structure. This matches an upper bound derived in an earlier paper.

1. IMPLICIT DATA STRUCTURES FOR DICTIONARIES

It is well-known that a dictionary of n elements can be implemented as a balanced tree, so that insert, delete, and search operations take $\Theta(\log n)$ time.[1] However such an implementation uses $\Theta(n)$ pointers, in addition to the locations needed to store key values. How important is this additional space in realizing these operation times? Several recent papers [MS, F1, F2] have sought to explore this issue by examining implicit data structures. An *implicit data structure* [MS] is a data structure that uses just a constant number of locations beyond those needed for the key values. The elements may be arranged in the first n locations of an array, with one additional location holding the number n. Any information about the relationship among the elements must be encoded in the way the elements are arranged. The information embodied in the arrangement of the elements may then be used in searching the structure.

Let S be the number of comparisons necessary for a search, and C the number of locations from which data must be moved to perform a change, i.e., a deletion followed by an insertion. Munro and Suwanda [MS] have shown that any implicit data structure that is stored in a fixed partial order must have a product of $S \times C$ no smaller than n. It is possible to do better if the implicit data structure is allowed to satisfy any partial order from a set of partial orders. Munro and Suwanda have exploited this approach with their rotated list structure, with which they are able to achieve $O(\log n)$ search times and $O(n^{1/2} \log n)$ insertion and deletion times. This approach has been exploited further by the present author in the design of a recursively rotated list [F1, F2]. This structure supports a search time of no more than $2 \log n + O(\sqrt{\log n})$ element comparisons, and insertion and deletion times that are $O(2^{\sqrt{2 \log n}} (\log n)^{3/2})$. The update times are a considerable improvement, since

---

The first portion of this research was supported in part by the National Science Foundation under Grant MCS-7909259.

[1] Logarithms to the base 2 will be denoted as log, and logarithms to the base e will denoted as ln.

they are $o(n^\varepsilon)$ for any constant $\varepsilon > 0$. But the question remains: Is this the best possible?

In this paper we wish to establish lower bounds for this problem. Understandably, determining the complexity of update in an implicit data structure in an unrestricted setting appears quite difficult. We will thus consider a somewhat restricted case, in which the structure is constrained to satisfy what we call a recursively rotated order. The recursively rotated lists satisfy these orders, so that we may determine a lower bound on how well these lists may perform.

We consider a structure of fixed size that satisfies a recursively rotated order, and bound the number of data swaps necessary to insert an arbitrary element in exchange for either the maximum or minimum element. We show that $\Omega(2^{\sqrt{2 \log n}} (\log n)^{1/2})$ swaps are required in worst case. This matches, to within a constant multiplicative factor, the number of swaps used in recursively rotated lists [F1,F2].

A complete version of this paper appears in [F3].

2. RECURSIVELY ROTATED ORDERS

In this section we introduce an order that we term a recursively rotated order. Sorted lists, rotated lists [MS] and recursively rotated lists [F1,F2] satisfy such orders. We identify an operation termed an exchange that captures the complexity of an update in a set maintained in a recursively rotated order. Then, as a function of the cost of an exchange, we characterize the particular structure of the recursively rotated order that gives us a bound on the number of elements in the set. We first define a recursively rotated order, which may be viewed as a relaxation of a total order.

Let S be a set of fixed size n, drawn from a universe with a total order. A *recursively rotated order* is a relation on S satisfying the following properties:

1. If $n = 1$, the relation is empty.
2. Otherwise, the set is partitioned into $m > 1$ subsets *(blocks)* $S_0, S_1, \cdots, S_{m-1}$ of sizes $n_0, n_1, \cdots, n_{m-1}$ such that:
   a. The relation among elements in the same subset is a recursively rotated order.
   b. The relation among elements in different subsets is described as follows: There is a value $p \in \{0,1, \cdots, m-1\}$ such that
      i. $x < y$ for all $x \in S_i$ and all $y \in S_{(i+1) mod\ m}$, for all $i \in \{0,1, \cdots, m-1\} - \{p, (p+1) mod\ m\}$
      ii. For all $z \in S_p$, either $z < y$ for all $y \in S_{(p+1) mod\ m}$ or $z > x$ for all $x \in S_{(p-1) mod\ m}$.

A recursively rotated order is *fixed* if the sizes of all subsets and recursively defined subsets are fixed. Note that the values of p are not fixed, so that a fixed recursively rotated order admits of a large number of fixed partial orders. A recursively rotated order is shown in Figure 1, in which the smallest circles represent single elements. A set satisfying this order is shown in Figure 2. The relations described by 2.b.i are shown with solid arrows, and the relations described by 2.b.ii are shown with dotted arrows. A recursively rotated order will be assumed fixed unless specified otherwise.

We briefly indicate how one might search in a set organized in a recursively rotated order. A more complete discussion may be found in [F1,F2]. For each block S_i in the set, a representative element a_i may be chosen. Because of the order imposed by 2.b, we will have

i. $a_i < a_{(i+1) mod\ m}$ for all $i \in \{0,1, \cdots ,m-1\} - \{p,(p+1) mod\ m\}$

ii. either $a_p < a_{(p+1) mod\ m}$ or $a_p > a_{(p-1) mod\ m}$

These representatives thus form a sorted list that has been circularly shifted. Search procedures that are modifications of binary search may be used to identify the position p, and then to identify the appropriate block in which to search recursively. If the sizes of the blocks are chosen appropriately (see [MS,F1,F2]), then the search times will be $O(\log n)$. Thus while a recursively rotated order is much less stringent than a total order, it is still possible to search efficiently a set that is organized according to it.

The recursively rotated order as we have defined it is static with respect to the size of the subsets. In an environment in which a set can grow and shrink due to insertions and deletions, this may not be sufficient. The definition can be extended to a dynamic setting by considering a sequence of recursively rotated orders, each describing a set of size one greater than the previous, and differing only by the addition of one more element in some subset. However, it is not necessary to consider sets of varying sizes in deriving our bounds. We can determine the complexity of updating the set by considering an exchange operation, in which an element is simultaneously inserted and one is deleted.

We shall distinguish between two types of exchanges that are natural for these structures. A *special exchange* on a set maintained in a recursively rotated order is an operation that either

1. inserts an element smaller than any in set S in exchange for the largest element, or

2. inserts an element larger than any in S in exchange for the smallest element.

A special exchange is easily performed, since only one data swap is required, replacing the outgoing element with the incoming one. A *general exchange* (henceforth, an *exchange*) is an operation that either

1. inserts an element of arbitrary rank with respect to S in exchange for either specifically the largest or the smallest element, or

2. inserts an element either larger than or smaller than any element in exchange for a specified element of arbitrary rank in S.

It takes more work to maintain the specified recursively rotated order when a general exchange is performed. This type of exchange may be implemented in the following fashion. Let S_i be the block containing the outgoing element, and S_j the block that will receive the incoming element. The block that will receive the incoming element will be the one identified by an unsuccessful search for that element. Assume that the exchange is of type 1, and $i \neq j$. The exchange may be accomplished by performing a type 1 exchange in S_j to extract the largest element. While the next block is not S_i, perform a special exchange in the next block, exchanging the extracted element (now smaller than any in this block) for the largest. When block S_i is reached, the displaced element will be of arbitrary rank with respect to the block, so that an exchange of type 1 must be performed in S_i. It does not matter in which direction around the sequence of blocks the displaced elements are moved. Moving in the direction of decreasing indices would require extracting smallest elements rather than largest. Thus the direction involving the fewest intervening blocks may be chosen. An exchange of type 2 may be handled in an analogous fashion.

We first describe the cost of an exchange in terms of the number of swaps required within S_i and S_j.

Lemma 1. Let S be a set arranged in a recursively rotated order. Let an exchange be performed such that the block S_i containing the outgoing element requires w_i swaps in worst case for an exchange, the block S_j (with $i \neq j$) that will receive the incoming element requires w_j swaps in worst case for an exchange, and the number of intervening blocks (in the shorter direction) is b. In worst case, $w_i + w_j + b$ swaps are required for the exchange.

Proof. The outgoing element in S_i must be replaced by an appropriate element. Suppose that the outgoing element is the only element replaced in S_i. To recover the required order in S the replacement should be either the largest element from $S_{(i-1) mod\ m}$ or the smallest element from $S_{(i+1) mod\ m}$. Now either the outgoing element is the smallest or largest in S (and hence in S_i) or it is of arbitrary rank in S (and without loss of generality also in S_i). If the former case holds, then the replacement element may be of arbitrary rank in S_i; if the latter holds then the replacement element may be the new smallest or new largest in S_i. Thus in either case this exchange will require w_i swaps in worst case. The replacement element from $S_{(i \pm 1) mod\ m}$ must be replaced, and the order allows this to be done in one swap, using the appropriate element from $S_{(i \pm 2) mod\ m}$. Thus it may be seen that a total of b data swaps are required to reestablish the order for S in the intervening blocks. The replacement element taken from S_j must of course be replaced by the insertion element. This is equivalent to an exchange in S_j, which requires w_j swaps in worst case. The required bound is achieved in this case.

Suppose there are other elements besides the outgoing element that are replaced in S_i. To preserve the required order in S, all replacements must be of consecutive ranks in S. Thus all replacements either come from the same block $S_{(i \pm 1) mod\ m}$ or from several consecutive blocks starting with $S_{(i \pm 1) mod\ m}$. We consider the first case, with the latter case handled by a similar argument. The replacement elements must be replaced, and the the displaced elements must displace others. These requirements will propagate around the sequence to the block S_j. It can be shown by an inductive argument that such a multiple exchange such as those performed in S_i and S_j will never require fewer swaps than a general exchange. Thus the lemma follows. ▪

We say that a recursively rotated order is *standard* if

1. whenever $m = 2$ and one block consists of a single element, the other block also consists of a single element, and
2. the relation among elements in the same block is a standard recursively rotated order.

We limit ourselves to considering standard recursively rotated orders for the following reason. Suppose $m = 2$ and only one of the blocks consists of a single element. The single value may have any possible relation with the elements in the other block. A search can be forced to examine both the single element and the other block. An exchange will involve either a swap with the element or an exchange in the other block. However the exchange will first require a search to find the element to be replaced. Thus there must be work done in both blocks in implementing an exchange, even though this does not show up in the count of swaps performed. Having a nonstandard order will therefore not help improve the operation times. A recursively rotated order will be assumed to be standard unless specified otherwise.

Given a set that requires w swaps in worst case, we wish to characterize the sizes and arrangement of blocks that maximizes the number of elements in the set. We provide a characterization in terms of not only w, but also the number of blocks m and the number of swaps t required in the largest block. Let $m = 2r+s$, where $s \in \{0,1\}$. The arrangement will be specified by a sequence of the number of swaps allowed for an exchange in each of the blocks. We define this arrangement $A(w,t,m)$ as:

1. t^m if $2t+r-1 \le w$
2. $t^{w-2t+2}(t-1) \cdots (w-t-r+2)(w-t-r+1)^{w-2t+2+s}(w-t-r+2) \cdots (t-1)$ if $2t \le w < 2t+r-1$
3. $t(w-t) \cdots (w-t-r+2)(w-t-r+1)^{1+s}(w-t-r+2) \cdots (w-t)$ if $w < 2t$

We elucidate the above notation: In case 1, t^m means m consecutive blocks, with each block allowing t swaps for an exchange inside of it. In case 3, the arrangement is a block allowing t swaps, followed by a block allowing $t-w$ swaps, etc.

Let $B(\cdot)$ be a function that satisfies the following properties:

1. $B(i+c) + B(j-c) \ge B(i) + B(j)$ for $i \ge j \ge c \ge 1$.
2. $B(w')$ is an upper bound on the maximum number of elements that can be held in any structure requiring w' swaps for an exchange.

We show that the arrangement $A(w,t,m)$ maximizes the bound on the number of elements in the set in terms of the $B(\cdot)$ bounds on the blocks.

Theorem 1. Let S be a set arranged in a recursively rotated order in which there are $m = 2r+s$ blocks, $s \in \{0,1\}$. Let w be the maximum number of data swaps needed to perform an exchange in S, and let t be the maximum number of swaps required in performing an exchange in any one of the blocks. Then S will realize the largest bound on the number of elements in terms of $B(\cdot)$ bounds on its blocks if it consists of the $A(w,t,m)$ arrangement.

Proof. We examine each range in the definition of $A(w,t,m)$ separately. We shall examine the third case here, leaving the other two to [F3].

case 3. ($w < 2t$) One block requires t data swaps in worst case. For each other block, the maximum number of data swaps that may be used is $w-t-b$ if there are b intervening blocks, by Lemma 1. This gives the arrangement as required. ▪

3. A BOUNDING FUNCTION

In this section we present a function $M(\cdot)$ that will bound, as a function of the number of swaps required for an exchange, the number of elements in a set arranged in a recursively rotated order. We shall first prove several simple properties about the function, and then show that it indeed serves the purpose required by the function $B(\cdot)$ in section 2. Let

$$M(0) = 0,\ M(1) = 2,\ M(2) = 6, \text{ and}$$
$$M(w) = \sum_{i=1}^{w} M(\lceil (i+1)/2 \rceil) \qquad \text{for } w > 2.$$

As we shall verify later, the first three values of $M(\cdot)$ are the exact maximum number of elements that the corresponding number of swaps will support. It may be determined that all other values are strict upper bounds.

Proofs of the following properties may be found in [F3].

Lemma 2. For $i > 0$

$$M(i+1) - M(i) \geq M(i) - M(i-1) \ ▪$$

Lemma 3. For $i \geq j \geq c \geq 1$

$$M(i+c) + M(j-c) \geq M(i) + M(j) \ ▪$$

We are now ready to show that $M(\cdot)$ is indeed an appropriate function to choose. We show that for any combination of w, t and m as in Theorem 1, $M(w)$ is an upper bound on the sum of $M(\cdot)$ applied to sizes of blocks in the arrangement $A(w,t,m)$.

Lemma 4. Let w, t, m, r and s be integers, with $m = 2r+s$, $s \in \{0,1\}$, $w \geq t+r$ and $w > 2$. Then

$$M(w) \geq \sum_{i \in A(w,t,m)} M(i)$$

Proof. We consider the various cases in the definition of $A(w,t,m)$. We present one here, and leave the rest for [F3].

case 1: ($2t+r-1 \leq w$)
case 1a: ($2r+1 \leq w$) From the definition of $M(w)$

$$M(w) = \sum_{i=1}^{w} M(\lceil (i+1)/2 \rceil)$$

$$\geq \sum_{i=w-2r}^{w} M(\lceil (i+1)/2 \rceil)$$

$$= M(\lceil (w-r+1)/2 \rceil) + \sum_{j=1}^{r} M(\lceil (w-r+1+j)/2 \rceil) + M(\lceil (w-r+1-j)/2 \rceil)$$

$$\geq (2r+1) M(\lceil (w-r+1)/2 \rceil) \qquad \text{by Lemma 3}$$

$$\geq m\ M(t)$$

since $2r+1 \geq m$, and $2t+r-1 \leq w$ implies $t \leq (w-r+1)/2$. ▪

Theorem 2. Let $N(w)$ be the largest number of elements stored in a structure so that no more than w data swaps are required for an exchange. Then

$$M(w) \geq N(w)$$

Proof. The proof is by induction on w. If $w = 1$, then the largest structure never requiring more than one data swap has two elements. This follows by considering various cases. Suppose the structure had three elements, arranged as three blocks consisting of single elements, e.g., 5,7,9. Inserting 6 in exchange for the largest element would require two swaps. Suppose the structure had two blocks, each consisting of more than one element, e.g., 5,7 and 3,9. Again an exchange of 6 for the largest would require two swaps. Hence $M(1) = N(1) = 2$. If $w = 2$, it may be verified in a similar fashion that the largest structure requiring no more than two swaps will be organized as three blocks, each containing two elements. Thus $M(2) = N(2) = 6$.

For $w > 2$, we assume that $M(w') \geq N(w')$, for all $w' < w$. Thus $M(\cdot)$ satisfies the requirements of the bounding function $B(\cdot)$ used in Theorem 1. From Theorem 1, we have that the largest bound on the number of elements in terms of the $B(\cdot)$ bounds on its blocks will be realized when the blocks are in the $A(w,t,m)$ arrangement. From Lemma 4,

$$M(w) \geq \sum_{i \in A(w,t,m)} M(i)$$

Thus $M(w)$ is an upper bound on $N(w)$. ▪

4. LOWER BOUND ON THE NUMBER OF SWAPS

In this section we determine the asymptotic behavior of our bounding function $M(\cdot)$. We then invert the function to get a lower bound on w, the number of swaps required for an exchange, in terms of the number of elements in the set. We finally compare this result with an upper bound that follows from the recursively rotated list structure in [F1,F2]. We thus establish the complexity of updates in our model.

Lemma 5. Let $M(\cdot)$ be the function defined in the previous section. Then

$$M(w) = 2^{(\log w)^2/2 - \log w \log\log w + \Theta(\log w)}$$

Proof. It can be seen that $M(\cdot)$ has essentially the same asymptotic growth rate as $f(\cdot)$, where

$$f'(x) = f(x/2 + u)$$

for some constant u. We will show that for large enough x

$$f(x) = e^{a(\ln x)^2 - b(\ln x)(\ln\ln x) + c \ln x}$$

for fixed constants a, b and c. We verify that the above expression is the solution. First

$$f'(x) = f(x)\, 2a(\ln x)/x\ (1 + \Theta((\ln\ln x)/\ln x))$$

Then using the approximation $\ln(1+\varepsilon) \approx \varepsilon$ for small ε, it is not hard to show that

$$f(x/2 + u) = f(x)\, e^{(b \ln\ln x - 2a \ln x - c)\ln 2 + \Theta(1)}$$

for large enough x. With $a = 1/(2 \ln 2)$ and $b = 1/\ln 2$, we have

$$f(x/2 + u) = f(x)\, (\ln x)/x\ (2^{-c}\ \Theta(1))$$

Since c may be adjusted as needed, the expression does satisfy the recurrence. Thus

$$f(x) = e^{((\ln x)^2/2 - (\ln x)(\ln\ln x) + c \ln 2 \ln x)/\ln 2}$$

$$= 2^{(\log x)^2/2 - (\log x)(\log\log x) + c' \log x}$$

where $c' = c + \log\log e$. Since $f_1(x)$ and $f_2(x)$ are $\Theta(f(x))$, the result follows. ▪

We may now invert $M(w)$ to get a lower bound on w, the number of swaps.

Theorem 3. The number w of data exchanges required in worst case to perform an exchange in a set maintained in a recursively rotated order is

$$w \geq c'\, 2^{\sqrt{2 \log n}}\ (\log n)^{1/2}$$

for some constant c'.

Proof. From Theorem 2 and Lemma 5, we have

$$N(w) \leq M(w) = 2^{(\log w)^2/2 - \log w \log\log w + \Theta(\log w)}$$

Taking logarithms, and multiplying by 2, we get

$$2 \log N \leq (\log w)^2 - 2 \log w\ \log\log w + \Theta(\log w)$$

$$\leq (\log w - \log\log w + d)^2$$

for some constant d. Thus

$$\log w \geq \sqrt{2 \log N} + \log\log w - d$$

$$w \geq 2^{\sqrt{2 \log N}}\ \log w\ \ 2^{-d}$$

This gives the result as claimed. ▪

The reader may note that the lower bound in Theorem 3 does not quite match the upper bound of $O(2^{\sqrt{2 \log n}}\ (\log n)^{3/2})$ necessary for the best structure cited in section 1. The reason is that we are not counting the work necessary to find the appropriate elements to swap. Indeed, if we review the analysis of the performance of this structure in [F1,F2], we discover that the number of swaps performed is smaller by a multiplicative factor of $\Theta(\log n)$.

Corollary 1. The complexity of an exchange in a set maintained in a recursively rotated order, expressed in terms of data swaps, is

$$\Theta(2^{\sqrt{2\log n}}\,(\log n)^{1/2}) .$$

Proof. The lower bound follows from Theorem 3. For the upper bound, consider the expandable recursively rotated list structure in [F1,F2]. Let $E(k)$ be the number of data swaps required for a special exchange in a list of k levels. A general exchange will require $H(k)$ data swaps, where

$$H(1) = 1$$

$$H(k) \le 2\,H(k-1) + 2^k\,E(k-1) \qquad \text{for } k > 1 .$$

We know that $E(k) = 1$ for all $k > 0$. The solution of the recurrence is then $O(k\ 2^k)$. Since $k = \sqrt{2\log n} + \Theta(1)$, the upper bound follows. ▪

REFERENCES

[F1] G. N. Frederickson, Implicit data structures with fast update, *Proc. 21st IEEE Symp. on Foundations of Computer Science*, Syracuse, New York (1980) 255-259.

[F2] G. N. Frederickson, Implicit data structures for the dictionary problem, to appear in *J.ACM*.

[F3] G. N. Frederickson, Recursively rotated orders and implicit data structures: a lower bound, Department of Computer Science technical report CS-82-01, Pennsylvania State University (1982).

[MS] J. I. Munro and H. Suwanda, Implicit data structures for fast search and update, *J. Comp. Sys. Sci. 21* (1980) 236-250.

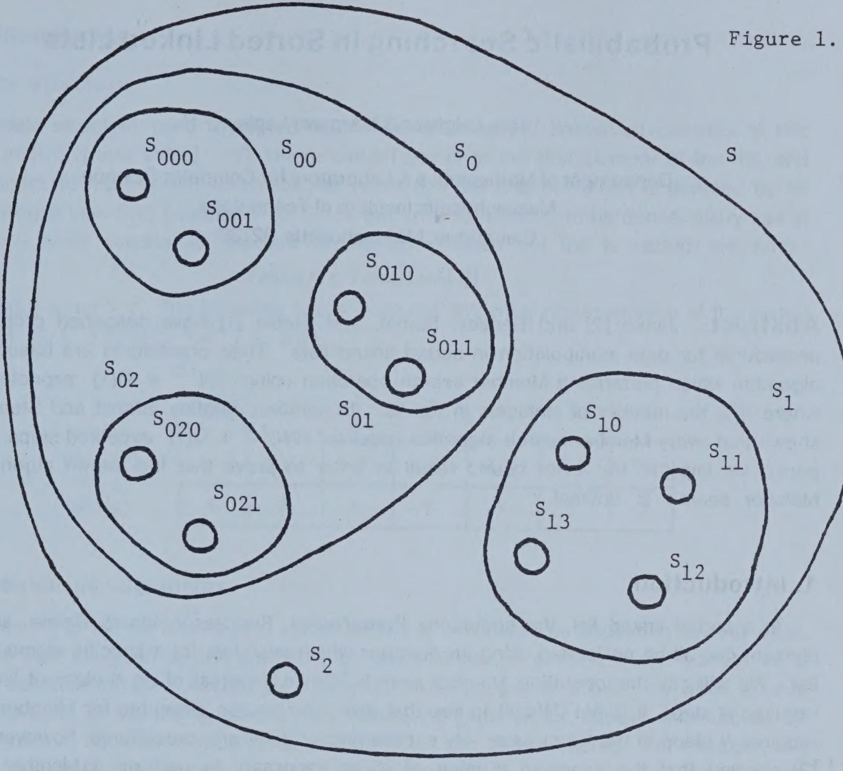

S
S_0
S_00
S_000
S_001
S_010
S_011
S_01
S_02
S_020
S_021
S_1
S_10
S_11
S_13
S_12
S_2

Figure 1.

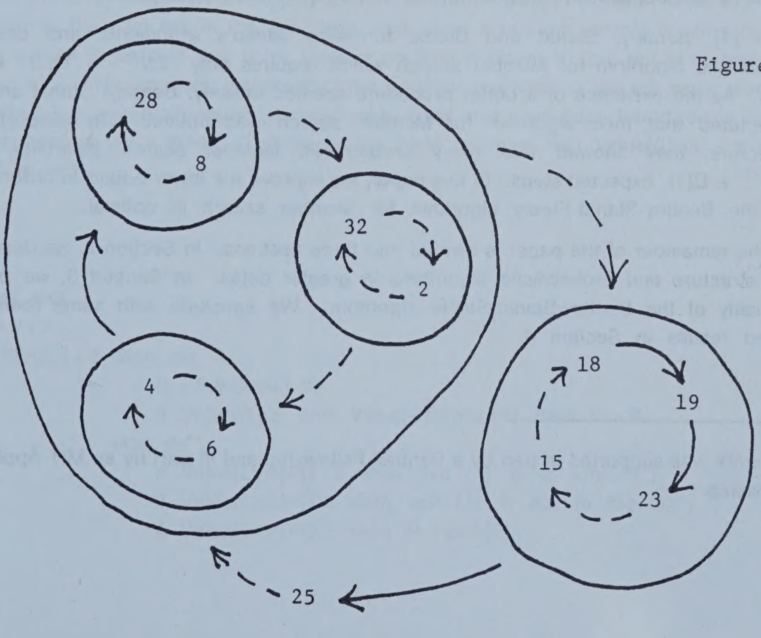

28
8
32
2
4
6
18
19
15
23
25

Figure 2.

Probabilistic Searching in Sorted Linked Lists

Tom Leighton & Margaret Lepley

Department of Mathematics & Laboratory for Computer Science
Massachusetts Institute of Technology
Cambridge, Massachusetts 02139

Abstract: Janko [2] and Bentley, Stanat, and Steele [1] have described probabilistic procedures for data manipulation in sorted linked lists. Their procedures are based on an algorithm which performs a Member search operation using $2N^{1/2} + O(1)$ expected steps where N is the number of elements in the list. In addition, Bentley, Stanat and Steele have shown that every Member search algorithm requires $(2N)^{1/2} + \Omega(1)$ expected steps. In this paper, we improve the lower bound result in order to prove that the known algorithm for Member search is optimal.

1. Introduction

In a sorted linked list, the operations Predecessor, Successor, Insert, Delete, and Last element can all be performed using an operator which searches for a specific element in the list. We will call this operation *Member search*. Since traversal of an N-element linked list requires N steps, it is not difficult to see that any *deterministic* algorithm for Member search requires N steps in the worst case. By considering *probabilistic* procedures, however, Janko [2] showed that the *expected* number of steps necessary to perform a Member search operation is substantially less than that required in the worst case.

In [1], Bentley, Stanat and Steele formalize Janko's arguments and describe a probabilistic algorithm for Member search which requires only $2N^{1/2} + O(1)$ expected steps. As the existence of a better procedure seemed unlikely, Bentley, Stanat and Steele conjectured that their algorithm for Member search was optimal. In support of this conjecture, they showed that *every* probabilistic Member search algorithm requires $(2N)^{1/2} + \Omega(1)$ expected steps. In this paper, we improve the lower bound in order to prove that the Bentley-Stanat-Steele algorithm for Member search is optimal.

The remainder of the paper is divided into three sections. In Section 2, we describe the data structure and probabilistic algorithms in greater detail. In Section 3, we prove the optimality of the Bentley-Stanat-Steele algorithm. We conclude with some remarks and related results in Section 4.

This work was supported in part by a Bantrell Fellowship and in part by an MIT Applied Math Fellowship.

2. Preliminaries

2a) Data structure

The data structure used to represent an N-element sorted linked list consists of two arrays, Link[$0...N$] and Value[$1...N$], where Link[0] points to the first element of the list, and Link[J] points to the element which follows Value[J]. The end of the list is denoted by an element whose Link field contains -1. Since the array is required to be dense, every cell in Value[$1...N$] must contain an element of the list. Since the list is sorted, we have

$$\text{Value}[J] \leq \text{Value}[\text{Link}[J]]$$

when Link[J] is not -1. The following figure illustrates the array representation of the sorted linked list $\langle 2.1,\ 3.5,\ 6.2,\ 7.4,\ 7.9,\ 9.6\rangle$.

J	0	1	2	3	4	5	6
Value[J]		3.5	6.2	9.6	2.1	7.4	7.9
Link[J]	4	2	5	-1	1	6	3

2b) Probabilistic algorithms

An element of the list is accessed via an index key which can be chosen in several ways. The index may be given as the link from the predecessor, a predetermined integer, or a randomly selected integer. Access via a predetermined integer is of use only when access to the beginning of the list is needed. By combining probabilistic access with access by predecessor link, however, a wide range of algorithms can be considered. For example, the following pseudo-Pascal program searches for the element E, using the order of operations specified by the array Step. When Step[J] is zero, a random sample occurs. Otherwise the program follows the next link in the list. Note that when a random sample is chosen, the position in the list is updated only if the random position is closer to (but not at or beyond) the location of E. This strategy insures that the updated positions in the list never worsen and that when E is eventually found, its predecessor will also have been found (since E will have been reached via a link). (This particular code assumes that Value[0] is $-\infty$ and Value[-1] is ∞.)

```
P: = 0
J: = 0
do until exit
    J: = J + 1
    if Step[J] = 0 then do
                R: = Random(1,N)
                if Value[R]<E and Value[R]>Value[P] then P: = R
            else do
                if Value[Link[P]] = E then exit (*E is at Link[P]*)
                if Value[Link[P]]>E then exit (*E is not in the list*)
                if Value[Link[P]]<E then P: = Link[P]
```

It is easy to modify the preceding algorithm for Member search to perform Predecessor, Successor and Last element. In addition, the algorithm can be modified to perform Insert and Delete. For example, if E does not appear in the list, then the search will end when the position it ought to occupy in the list has been found. The element can then be inserted by placing it at index $N+1$ and changing the links. A Delete is more complex. First the element to be deleted is found and removed from the list. There is now an index in the array which contains no element of the list. To remedy this situation the element at index N is moved into the open spot. This of course requires another search operation to find the Predecessor of the element at index N. Therefore a Delete requires two Member searches if the arrays are to remain dense.

2c) Mathematical model

In studying the random algorithm presented above, we will assume that we are searching for the largest element in the list. This is at least as difficult as seaching for a random element in the list, since the operations we are using only allow movement in the direction of the links. The program is then a black box which must be told the order of random and deterministic operations. This behavior can be modelled by the following probabilistic game due to Bentley, Stanat, and Steele [1].

Definition: The *probabilistic game G-D* involves two integers, i and N. The value of N remains fixed throughout the game, and the value of i is originally N. The goal of the player is to reduce the value of i to zero in the minimum expected number of steps. A step consists of performing one of the following two operations-

*D(for Decrement): If $i>0$, then replace i by $i-1$.

*G(for Guess): Choose j to be an integer uniform from $1 \ldots N$ and replace i by j if $j<i$. The value of i is unknown to the player, except at the beginning of the game when $i=N$, and at the end when he is notified that i has reached zero. (In this game the value i represents the distance from the current element in the list – denoted by P in the code above – to the end of the list.)

A strategy or sequence of operations will be denoted by a character string σ, composed of G's and D's, to be performed in order from left to right. A sequence is said to be *complete* if it contains at least N D's. Note that operations written after the first N D's are superfluous and need not appear. For convenience, however, we will often end complete sequences with D^N.

A complete sequence will always reach $i=0$ and terminate the game after some number of steps t. The expected termination point for a complete sequence σ is denoted by

$$E(\sigma) = \sum_{j=1}^{\infty} j \Pr(t=j) = \sum_{j=0}^{\infty} Q_j(\sigma)$$

where $Q_j(\sigma) = \Pr[t>j]$. The object is to minimize $E(\sigma)$ over all complete sequences σ. We denote the minimum by $S(N)$.

2d) Previous results

Janko first investigated the effects of random sampling in a sorted linked list. He restricted his analysis to the class of strategies of the form G^kD^N, for some k. Within this restricted class he discovered the optimal value for k.

Theorem 1: *For any integer* $k \geq 0$, $E(G^rD^N) \leq E(G^kD^N)$ *where* $r = [N^{1/2} - 1 - 1/(24N^{1/2})]$.

Proof: From the definition,

$$E(G^kD^N) = \sum_{j=0}^{\infty} Q_j(G^kD^N).$$

Since the first k operations are Guesses the game cannot end there, so $Q_j(G^kD^N) = 1$ for $j = 0,..,k$. The probability of terminating after d Decrements is $(N-d)^kN^{-k}$, since all the Guesses must be larger than d. Therefore

$$\begin{aligned} E(G^kD^N) &= k + 1 + \sum_{d=1}^{N}(N-d)^kN^{-k} \\ &= k + 1 + N^{-k}\sum_{d=0}^{N-1}d^k \\ &= k + 1 + N/(k+1) - 1/2 + k/(12N) + O(k^2/N^2) \end{aligned}$$

The minimum occurs when

$$1 + 1/(12N) \sim N/(k+1)^2 \text{ or } k \sim N^{1/2} - 1 - 1/(24N^{1/2}) \quad \square$$

Theorem 1 provides an upper bound of $S(N) \leq 2N^{1/2} - 1/2 + 1/(12N^{1/2}) + O(1/N)$ expected operations for the G-D game. Recently, Bentley, Stanat, and Steele found a lower bound for the game. In particular, they showed that if the player was allowed to know the value of i at all times, then the optimal strategy requires $\sim(2N)^{1/2}$ expected steps. Hence, $S(N) = \Theta(N^{1/2})$ where the constant is between $2^{1/2}$ and 2. We will now show that the lower bound for the general G-D game can be improved to $\sim 2N^{1/2}$, which is tight.

3. General Lower Bound

When a sequence ω is not complete, the value of i after the operations in ω have been performed may remain undetermined. Instead of knowing the exact value of i we define a probability vector $P(\omega)$ so that

$$P_j(\omega) = \text{Prob}[i > j \text{ after executing the sequence } \omega].$$

We can see from the definition that $P_j(\omega) \leq P_{j+1}(\omega)$. Moreover the vector can be computed for any sequence ω.

Lemma 1: *For any sequence* $\omega = G^{a_1}D^{b_1}...G^{a_s}D^{b_s}$, *with* $b = b_1 + ... + b_s$ *and* $a = a_1 + ... + a_s$

$$P_j(\omega) = \begin{cases} (N-j-b)^{a_1}(N-j-b+b_1)^{a_2}...(N-j-b_s)^{a_s}N^{-a} & \text{for } j < N-b \\ 0 & \text{for } j \geq N-b \end{cases}$$

Proof: During each block of Guesses, G^{a_m}, the value i must remain above j plus the number of D's which are still to be performed, $b_m + ... + b_s$. The probability that all the Guesses in the block are between $j + b_m + ... + b_s + 1$ and N is $(N-j-b_m-...-b_s)N^{-a_m}$ $\square$

These probabilities are important in determining the optimal strategy. The following lemma states one of the most useful properties of this vector.

Lemma 2: *For any sequence* ω, $P_j(\omega D)/P_0(\omega D) \leq P_j(\omega)/P_0(\omega)$ *for* $0 \leq j \leq N$.

Proof: Each ratio is a product of terms of the form

$$[(N-j-b_i-\ldots-b_s)/(N-b_i-\ldots-b_s)]^{a_i}.$$

The sequence ωD has one more D in the last block than ω, so b_s increases by one in ωD. When $P_0(\omega D)>0$ a comparison of these ratios, letting $K=N-b_i-\ldots-b_s$, shows

$$[(K-j-1)/(K-1)]^{a_i} < [(K-j)/K]^{a_i}.$$

If $P_0(\omega D)=0$, we define the ratio to be zero and the inequality still holds. □

Our goal in this section is to prove that the optimal strategy has the form G^iD^N. First we analyze the effect of minor variations in strategy on the expected number of operations. Then we will show that if a small variation improves the strategy then a larger change could mean even more improvement. The two sequences which we will compare first are $\sigma = \omega DG^kD^N$ and $\sigma^* = \omega G^kD^N$. The only difference between σ and σ^* is the position of the block G^k. The following lemma gives a method for comparing these two strings.

Lemma 3: $E(\omega DG^kD^N) \leq E(\omega G^kD^N)$ *if and only if* $V_k(\omega) \leq 1$, *where*

$$V_k(\omega) = \begin{cases} P_1(\omega)/P_0(\omega) + \sum_{d=1}^{N} P_d(\omega)/P_0(\omega)\left[(N-d+1)^k - (N-d)^k\right]k^{-1}N^{-k} & \text{if } P_0(\omega)>0 \\ 0 & \text{if } P_0(\omega)=0 \end{cases}$$

Proof: Let σ and σ^* be defined as above. We would like to know when $E(\sigma)-E(\sigma^*) \leq 0$. The following values for $Q_m(\sigma)$ and $Q_m(\sigma^*)$ can be easily verified.

$$Q_m(\sigma) = \begin{cases} Q_m(\sigma^*) & \text{if } m \leq |\omega| \\ P_1(\omega) & \text{if } |\omega|+1 \leq m \leq |\omega|+k+1 \\ P_d(\omega)(N-d+1)^kN^{-k} & \text{if } d = m-k-|\omega|>1 \end{cases}$$

$$Q_m(\sigma^*) = \begin{cases} Q_m(\sigma) & \text{if } m \leq |\omega| \\ P_0(\omega) & \text{if } |\omega|+1 \leq m \leq |\omega|+k \\ P_d(\omega)(N-d)^kN^{-k} & \text{if } d = m-k-|\omega|>0 \end{cases}$$

Thus $E(\sigma)-E(\sigma^*) = \sum_{m=0}^{\infty}[Q_m(\sigma)-Q_m(\sigma^*)] \leq 0$ is equivalent to

$$(k+1)P_1(\omega) - kP_0(\omega) + \sum_{d=1}^{N} P_d(\omega)[(N-d+1)^k-(N-d)^k]N^{-k} - P_1(\omega) \leq 0.$$

Rearranging terms slightly gives

$$P_1(\omega) + \sum_{d=1}^{N} P_d(\omega)[(N-d+1)^k-(N-d)^k]N^{-k}k^{-1} \leq P_0(\omega) \qquad □$$

Combining Lemmas 2 and 3 enables us to extend a minor variation of the string into a more radical change. Specifically, if the last block of G's is more efficient when it is moved to the right one place, then it is best to remove the block of Guesses altogether.

Lemma 4: *For all* ω, *if* $E(\omega D G^k D^N) \leq E(\omega G^k D^N)$, *then* $E(\omega D^N) \leq E(\omega D G^k D^N)$.

Proof: If $E(\omega D^j G^k D^N) \leq E(\omega D^{j-1} G^k D^N)$ for some $j \geq 1$, then $V_k(\omega D^{j-1}) \leq 1$ by Lemma 3. By Lemma 2, we know that $V_k(\omega D^j) \leq V_k(\omega D^{j-1})$, and thus that $V_k(\omega D^j) \leq 1$. Thus, we can conclude by Lemma 3 that $E(\omega D^{j+1} G^k D^N) \leq E(\omega D^j G^k D^N)$. The proof of the lemma is completed by applying this process inductively □

It is now a simple matter to prove our main result.

Theorem 2: *There exists an integer* $r \geq 0$ *such that* $E(G^r D^N) \leq E(\sigma)$ *for every complete sequence* σ.

Proof: Let σ denote the shortest (in length) complete sequence which achieves $S(N)$. If σ is of the form $G^r D^N$, then we are done. Otherwise $\sigma = \omega D G^k D^j$, where G^k is the last block of Guesses and $j > 0$. By the optimality of σ, we know that $E(\omega D G^k D^j) \leq E(\omega G^k D^{j+1})$. Thus by Lemma 4, we know that $E(\omega D^{j+1}) \leq E(\omega D G^k D^j)$ which contradicts the minimality of σ □

Recalling Theorem 1, we find that the optimal value of r is $N^{1/2} - 1 - 1/(24N^{1/2}) + O(1/N)$ and thus

$$S(N) = 2N^{1/2} - 1/2 + 1/(12N^{1/2}) + O(1/N).$$

4. Remarks

At first glance, it might appear that our proof technique depends on not being able to Guess 0. This is not the case. It is also worth remarking that Theorem 2 can be extended to show that for all ω and ω', there is an integer $r > 0$ such that $E(\omega G^r D^N) \leq E(\omega\omega')$. Thus the best way to finish any initial sequence is by a block of Guesses followed by Decrements.

The optimal solution to the G-D game provides an optimal solution to the problem of finding the Last element in a sorted linked list. It does not necessarily provide the optimal solution to finding the *ith* element in the list, however. Although the best algorithm for the more general problem consists of a series a Guesses followed by a series of Decrements, the number of Guesses varies as a function of i. If i is not known (as is the case with Member search) then the best strategy is the same as with Last element except that one less Guess is performed.

The G-D game can be modified in several ways. One particularly interesting modification allows the player to use the information about when a random sample is successful (i.e., closer to the target). For example, suppose the black box containing i is connected to a light which turns on every time i is decreased. At first glance, it appears as though such information could be quite useful in planning when to stop Guessing and start Decrementing. (For instance, if the light flashed on for a series of early Guesses, then the player might be led to believe that the early Guesses were very good and thus that i had become very small.

Hence, the player might think it wise to start Decrementing early.) This is *not* the case, however, since if all the Guesses are required to be distinct, then it can be shown that the light does not add any useful information at all.

The G-D game might also be played with two different operations O_1 and O_2. It would be interesting to know what properties of O_1 and O_2 give an optimal sequence of the form $O_1^k O_2^m$, for some k and m. In addition to operators which act directly upon i, we might also consider comparison operations which compare i to a given input, n, and answer the question, "$i \leq n$?" If the compare operation $i \leq (2N)^{1/2}$ is added to G-D, then the optimal strategy is $O(N^{1/4})$ Guesses followed by a Compare, repeated until $i \leq (2N)^{1/2}$, ending with Decrements. This strategy uses $(2N)^{1/2} + O(N^{1/4})$ expected steps.

Acknowledgements

We would like to thank Gary Miller, Ron Rivest, Jim Saxe, and Mike Sipser for the many discussions of this problem and related ones. A special thanks to Jon Bentley for bringing this problem to our attention.

References

[1] J. L. Bentley, D. F. Stanat, J. M. Steele, "Analysis of a randomized data structure for representing ordered sets," *Proceedings Nineteenth Annual Allerton Conference on Communication, Control, and Computing*, pp. 364-372.

[2] W. Janko, "An insertion sort for keys with arbitrary key distribution," *ACM Transactions on Mathematical Software* 2, 2, June 1976, pp. 143-153.

DYNAMIC PROGRAMMING, LEVINSON ALGORITHM, AND LATTICE EQUALIZERS*

F. POLLARA and K. YAO
Department of System Science
University of California
Los Angeles, CA 90024

ABSTRACT

In this paper we shall use the dynamic programming technique to solve recursively the minimization of a generalized quadratic form and thus rederive the full general form of the Levinson Algorithm. It is well known that the partial correlation coefficients, which are the recursive solutions of the specialized Levinson Algorithm, have immediate interpretation in the lattice structure implementation of the linear predictor. We shall show that a corresponding set of coefficients, given by the recursive solutions of the general form of the Levinson Algorithm, has direct interpretation and application in the lattice structure implementation of a digital data equalizer.

I. INTRODUCTION

Linear data equalizers have been used in many digital data communication systems. In recent years, the lattice equalizer which efficiently implements the decorrelation of the received waveform, has been of great interest. The lattice structure offers several advantages over the conventional TDL structure. It is known that the lattice in diagonalizing the input covariance matrix, makes the equalizer insensitive to the spread of the eigenvalues of the matrix. This property is particularly important in adaptive equalization with regard to speed of covergence [1,2].

The development of lattice equalizers was based on that of lattice predictors, first used in speech processing and popularized by Makhoul [3]. The relevance of the partial correlation coefficients, which are the recursive solutions of the predictor form of the Levinson Algorithm in the lattice implementation of a linear predictor, is well known [3]. However, the full use of the general form of the Levinson Algorithm [4] in lattice equalizers, as compared to that of the special form of the Levinson Algorithm in lattice predictors, appears not to have been considered in any systematic manner.

In Section II, we use the "principle of optimality" of dynamic programming to rederive the general form of the Levinson Algorithm. This general recursive solution contains two parts, with one part given by the recursive solution of the linear predictor problem as an intermediate step. In Section III, we relate the lattice equalizer to the general form of the Levin-

---

* This work was partially supported by the Electronics Program of the Office of Naval Research.

son Algorithm. In particular, we show that in the implementation of the lattice equalizer, not only are the partial correlation coefficients, which are the solution of the lattice predictor used, but the other part of the solution of the general form of the Levinson Algorithm is also used explicitly in a direct manner.

II. LEVINSON ALGORITHM

In many estimation problems, the mean-square error is expressible as a generalized quadratic form

$$Q_p(\underline{u}_p) = u_o + \underline{c}'_p R_p \underline{c}_p - 2\underline{c}'_p \underline{u}_p, \tag{1}$$

where

$\underline{c}'_p = (c_1,\ldots,c_p) = (\underline{c}'_{p-1}, c_p)$ is a variable vector, $u_i, i=0,\ldots,p$ are fixed constants, $\underline{u}'_p = (u_1,\ldots,u_p)$, and R_p is a p×p symmetric Toeplitz matrix defined by

$$R_p = [r_{|i-j|}]_{i,j=1,\ldots,p} = \begin{bmatrix} R_{p-1} & J\underline{r}_{p-1} \\ \underline{r}'_{p-1}J & r_o \end{bmatrix} \tag{2}$$

with $\underline{r}'_p = (r_1,\ldots,r_p) = (\underline{r}'_{p-1}, r_p)$ and J is a rotation matrix with all zeros except on the second diagonal. Then the pre-multiplication of a row vector by J or the post-multiplication of a column vector by J reverses the order of the vector. Furthermore $JR_pJ = R_p$. We also note, for reasons which will be apparent soon, the argument of $Q_p(\cdot)$ in (1) is taken to be $\underline{u}_p$ and not the variable vector $\underline{c}_p$.

It is well known that the minimum of $Q_p(\underline{u}_p)$ in (1) is attained by $\underline{c}^*_p$ where

$$Q^*_p(\underline{u}_p) = \mathop{\mathrm{Min}}_{\underline{c}_p} Q_p(\underline{u}_p), \qquad \underline{c}^*_p = R_p^{-1}\underline{u}_p. \tag{3}$$

In practice, we are not only interested in the block solution $\underline{c}^*_p$ in (3), but are more interested in the recursive solution of this minimization problem [4].

First, we observe that (1) can be written recursively as

$$Q_p(\underline{u}_p) = c_p^2 r_o - 2c_p u_p + Q_{p-1}(\underline{u}_{p-1} - c_p J\underline{r}_{p-1}). \tag{4}$$

Then the minimization of $Q_p(\underline{u}_p)$ may be written as

$$\begin{aligned} Q^*_p(\underline{u}_p) &= \mathop{\mathrm{Min}}_{\underline{c}_p} Q_p(\underline{u}_p) \\ &= \mathop{\mathrm{Min}}_{c_p} \{c_p^2 r_o - 2c_p u_p + \mathop{\mathrm{Min}}_{\underline{c}_{p-1}} Q_{p-1}(\underline{u}_{p-1} - c_p J\underline{r}_{p-1})\} \\ &= \mathop{\mathrm{Min}}_{c_p} \{c_p^2 r_o - 2c_p u_p + Q^*_{p-1}(\underline{u}_{p-1} - c_p J\underline{r}_{p-1})\}. \end{aligned} \tag{5}$$

In Equation (5), we use the principle of optimality of dynamic programming in that once the "optimal policy" $Q^*_{p-1}(\cdot)$ for the order (p-1) problem has been found, $Q_p^*(\cdot)$ may be found by a simple scalar minimization [5]. By using

$$Q_p^*(\underline{u}_p) = r_o - \underline{c}_p^{*\prime}\underline{u}_p = r_o - \underline{u}_p^{\prime} R_p^{-1}\underline{u}_p, \tag{6}$$

(5) can be expressed as

$$\begin{aligned} Q_p^*(\underline{u}_p) &= \underset{c_p}{\mathrm{Min}}\ \{c_p^2 r_o - 2c_p u_p + u_o - (\underline{u}_{p-1} - c_p J\underline{r}_{p-1})^{\prime} R_{p-1}^{-1}(\underline{u}_{p-1} - c_p J\underline{r}_{p-1})\} \\ &= \underset{c_p}{\mathrm{Min}}\ \{c_p^2(r_o - \underline{r}_{p-1}^{\prime} J R_{p-1}^{-1} J\underline{r}_{p-1}) - 2c_p(u_p - \underline{u}_{p-1}^{\prime} R_{p-1}^{-1} J\underline{r}_{p-1}) + Q_{p-1}^*(\underline{u}_{p-1})\} \\ &= \underset{c_p}{\mathrm{Min}}\ \{c_p^2(r_o - \underline{b}_{p-1}^{*\prime}\underline{r}_{p-1}) - 2c_p(u_p - \underline{c}_{p-1}^{*\prime} J\underline{r}_{p-1}) + Q_{p-1}^*(\underline{u}_{p-1})\}, \end{aligned} \tag{7}$$

where we have defined

$$\underline{b}_p^* \triangleq R_p^{-1}\underline{r}_p, \underline{b}_p^{*\prime} \triangleq [b_1^p, \ldots, b_p^p]. \tag{8}$$

In order to solve (7), we are forced to solve (8), which is the solution of the "normal equations" of linear prediction [3] given by

$$R_p\underline{b}_p = \underline{r}_p. \tag{9}$$

Equation (9) in term is the solution of the "auxiliary minimization problem" of

$$Q_p(\underline{r}_p) = r_o + \underline{b}_p^{\prime} R_p \underline{b}_p - 2\underline{b}_p^{\prime}\underline{r}_p, \tag{10}$$

where

$$\underline{b}_p^{\prime} = [b_1, \ldots, b_p] = [\underline{b}_{p-1}^{\prime}, b_p].$$

While the desired block solution of (9) or (10) is given by (8), we are again more interested in the recursive solution. However, the expression in (10) is similar to that in (1) except for replacing $\underline{c}_p$ by $\underline{b}_p$ and $\underline{u}_p$ by $\underline{r}_p$. Thus, from (7), the minimization of $Q_p(\underline{r}_p)$ can be obtained immediately as

$$\begin{aligned} Q_p^*(\underline{r}_p) = \underset{b_p}{\mathrm{Min}}\ \{&b_p^2(r_o - \underline{b}_{p-1}^{*\prime}\underline{r}_{p-1}) \\ &- 2b_p(r_p - \underline{b}_{p-1}^{*\prime} J\underline{r}_{p-1}) + Q_{p-1}^*(\underline{r}_{p-1})\}. \end{aligned} \tag{11}$$

By minimizing the scalar quadratic form of b_p in (11), and denoting this optimum by k_p, we have

$$k_p = \frac{r_p - \underline{b}_{p-1}^{*\prime} J\underline{r}_{p-1}}{r_o - \underline{b}_{p-1}^{*\prime}\underline{r}_{p-1}} = \frac{r_p - \underline{b}_{p-1}^{*\prime} J\underline{r}_{p-1}}{Q_{p-1}^*(\underline{r}_{p-1})} \tag{12}$$

Direct substitution of (12) in (11) gives

$$Q_p^*(\underline{r}_p) = (1 - k_p^2) Q_{p-1}^*(\underline{r}_{p-1}). \tag{13}$$

Then the equivalent version of (6) for $Q_p^*(\underline{r}_p)$ in (13) yields

$$\underline{b}_p^{*\prime} = [(\underline{b}_{p-1}^* - k_p J\underline{b}_{p-1}^*)^{\prime}, k_p], \quad b_p^p = k_p. \tag{14}$$

Thus, (12)-(14) are used to find the full recursive solution of the auxiliary problem of (10).

Now, in the original problem in (7), let the optimum c_p be denoted by w_p. Then by the same technique used to obtain (12)-(14), we have

$$w_p = \frac{u_p - \underline{c}^{*\prime}_{p-1} J \underline{r}_{p-1}}{r_o - \underline{b}^{*\prime}_{p-1} \underline{r}_{p-1}} = \frac{u_p - c^{*\prime}_{p-1} J \underline{r}_{p-1}}{Q^{*}_{p-1}(\underline{r}_{p-1})} . \tag{15}$$

$$Q^{*}_{p}(\underline{u}_p) = Q^{*}_{p-1}(\underline{u}_{p-1}) - w_p^2 Q^{*}_{p-1}(\underline{r}_{p-1}). \tag{16}$$

$$\underline{c}^{*\prime}_{p} = [(\underline{c}^{*}_{p-1} - w_p J \underline{b}^{*}_{p-1})', \ w_p] \ , \quad c^{p}_{p} \equiv w_p . \tag{17}$$

Thus, (12)-(17) are needed in the recursive solution of the generalized quadratic form of (1). These results are summarized in Table 1 and a flow chart is given in Fig. 1.

III. LATTICE EQUALIZER

Consider a linear receiver designed for the optimal detection in the mean-square sense of discrete data in the presence of WGN and intersymbol interference. Let the sampled received waveform after the matched filter at time n be denoted by

$$x(n) = \sum_{i} u(i)\, s(n-i) + \nu(n), \tag{18}$$

where $u(i)$ represents the channel response, and $u(i) \neq 0$ for $i \neq 0$ causes ISI; $\nu(n)$ is a sequence of i.i.d. $N(0,\sigma_\nu^2)$ random variables; $s(i)$ is the data sequence which modulates the channel. Let the output of the linear equalizer be

$$y(n) = \sum_{i=1}^{p} c_i^p x(n-i+1) = \underline{c}_p' \underline{x}_p, \tag{19}$$

where

$$\underline{c}_p = (c_1^p, \ldots, c_p^p)' \ , \quad \underline{x}_p = (x(n), \ldots, x(n-p+1))'. \tag{20}$$

Under the mean-square criterion, for any fixed instant of time (by dropping the time variable n), we seek

$$\underset{\underline{c}_p}{\mathrm{Min}}\ E\{(y-s)^2\} = \underset{\underline{c}_p}{\mathrm{Min}}\ \{\underline{c}_p' R_p \underline{c}_p - 2\underline{c}_p' \underline{u}_p + u_o\}, \tag{21}$$

where

$$u_o = E\{s^2\}, \quad \underline{u}_p = E\{s\underline{x}_p\}, \tag{22}$$

$$R_p = E\{\underline{x}_p \underline{x}_p'\} = [E\{s^2\}\{\sum_{n=-\infty}^{\infty} u(n-j)u(n-i)\} + \sigma_\nu^2 \delta_{ij}] \ , \quad i,j=1,\ldots,p. \tag{23}$$

Clearly, the minimization of the generalized quadratic form in Section II is applicable to (21). Of course, if we consider a conventional TDL equalizer of the form given in Fig. 2, then the optimal set of TDL coefficients is given from (3) as $\underline{c}_p = R_p^{-1} \underline{u}_p$.

Now, consider the lattice structure implementation of the equalizer in (19). In order to state this problem appropriately, we need to consider the relationship of a conventional TDL one-step predictor and its lattice implementation. Consider the prediction of x(n) from a linear combination of $\{x(n-1),\ldots,x(n-p)\}$. Let the innovation of x(n) be denoted by

$$z(n) = x(n) - \sum_{i=1}^{p} b_i x(n-i) \tag{24}$$

If the sequence $\{x(n)\}$ is wide-sense stationary, then the orthogonality of z(n) to $\{x(n-1),\ldots,x(n-p)\}$ implies the optimal one-step predictor coefficients $\{b_1,\ldots,b_p\}'$ are given by

$$\underline{b}_p = (b_1,\ldots,b_p)' = R_p^{-1}\underline{r}_p, \tag{25}$$

which is a solution of the auxiliary problem. A TDL implementation of z(n) in (24) using the TDL coefficients in (25) is given in Fig. 3.

A lattice structure of this predictor problem is well known [3] and is given in Fig. 4. In particular, the orthogonalized version of the data $\underline{x}_p$ defined in (20) is given by

$$\underline{v}_p = (v_1,v_2,\ldots,v_p)' = L_p\underline{x}_p \tag{26}$$

where

$$L_p = \begin{bmatrix} 1 & & \\ -\underline{b}_1' J & 1 & 0 \\ & \cdot\;\cdot\;\cdot & \\ -\underline{b}_{p-1}' J & & 1 \end{bmatrix} = \begin{bmatrix} L_{p-1} & 0 \\ -\underline{b}_{p-1}' J & 1 \end{bmatrix} \tag{27}$$

Indeed each v_i in (26) is a backward version of the expression in (24) and is mutually orthogonal. Since L_p is invertible, we can express the output of the equalizer in (19) in terms of the orthogonal data $\underline{v}_p$ given by

$$y(n) = \underline{c}_p'\underline{x}_p = \underline{\tilde{w}}_p'\underline{v}_p, \tag{28}$$

where

$$\underline{\tilde{w}}_p' = \underline{c}_p' L_p^{-1} = (\tilde{w}_1,\ldots,\tilde{w}_{p-1},\tilde{w}_p) = (\underline{\tilde{w}}_{p-1}',\tilde{w}_p). \tag{29}$$

Furthermore,

$$\begin{aligned} \underline{c}_p' = \underline{\tilde{w}}_p' L_p &= (\underline{\tilde{w}}_{p-1}',\tilde{w}_p)\begin{bmatrix} L_{p-1} & 0 \\ -\underline{b}_{p-1}' J & 1 \end{bmatrix} \\ &= (\underline{\tilde{w}}_{p-1}' L_{p-1} - \underline{b}_{p-1}' J\tilde{w}_p,\ \tilde{w}_p) \\ &= ((\underline{c}_{p-1}' - \tilde{w}_p J \underline{b}_{p-1})',\ \tilde{w}_p). \end{aligned} \tag{30}$$

Since the optimal TDL coefficients $\underline{c}_p$ is given by the recursive solution of the generalized quadratic form in (17), then by comparing (30) with (17), it is clear that

$$\tilde{w}_i = w_i, \quad , \; i = 1,\ldots,p. \tag{31}$$

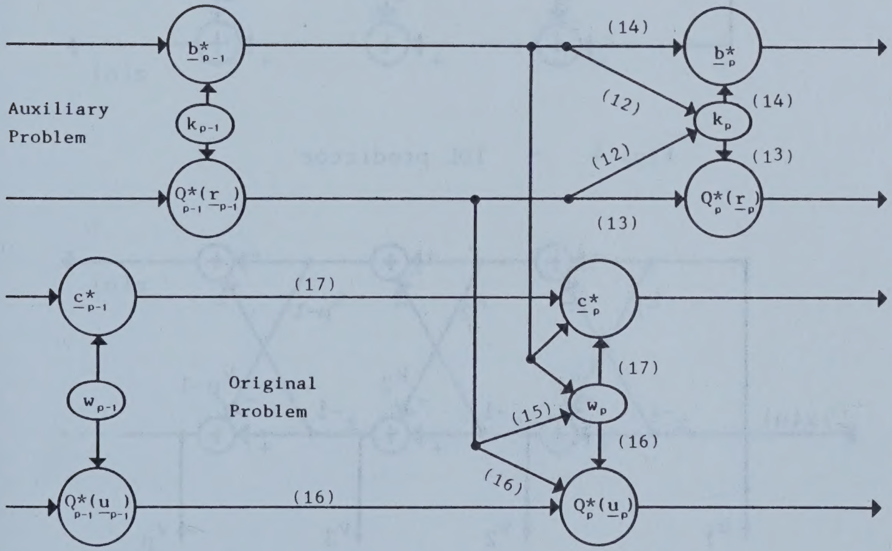
In summary, from (28) we see the conventional TDL equalizer given by $\underline{c}'_p \underline{x}_p$, is equivalent to the lattice equalizer $\underline{\tilde{w}}'_p \underline{v}_p$. Furthermore, the components of the new set of TDL coefficients $\underline{\tilde{w}}_p$ operating on the orthogonal data $\underline{v}_p$ are given explicitly by the w_i coefficients in the recursive solution of the generalized Levinson Algorithm in (12)-(17). A block diagram implementation of this lattice equalizer is given in Fig. 5. The approaches and results obtained here have shown to be useful in the quantitative understanding of the effect of finite precision operations in the lattice equalizer. These matters will be considered in details in other papers.

REFERENCES

1. E. Schichor, "Fast Recur. Est. Using the Lattice Structure," BSTJ, Vol. 61, Jan. 1982, pp.97-115.
2. E.H. Satorious and J.D. Pack, "Appl. of Least Sq. Lattice Algorithms to Adap. Equaliz.," IEEE Trans. on Commun., Feb. 1981, pp.136-142.
3. J. Makhoul, "Linear Prediction: A Tutorial Review," Vol.63, April 1975, pp.561-580.
4. N. Levinson, "The Wiener RMS Error Criterion in Filter Design and Prediction," J. Math. Phys., Vol.25, pp.261-278, 1947.
5. R. Bellman, "Dynamic Programming," Princeton Univ. Press, Princeton, NJ, 1957.

ORIGINAL PROBLEM	AUXILIARY PROBLEM
$Q^*_p(\underline{u}_p) = Q^*_{p-1}(\underline{u}_{p-1})-w_p^2 Q^*_{p-1}(\underline{r}_{p-1})$	$Q^*_p(\underline{r}_p) = Q^*_{p-1}(\underline{r}_{p-1})-k_p^2 Q^*_{p-1}(\underline{r}_{p-1})$
$w_p = \dfrac{u_p-\underline{b}^{*\prime}_{p-1} J\underline{u}_{p-1}}{Q^*_{p-1}(\underline{r}_{p-1})}$	$k_p = \dfrac{r_p-\underline{b}^{*\prime}_{p-1} J\underline{r}_{p-1}}{Q^*_{p-1}(\underline{r}_{p-1})}$
$\underline{c}^{*\prime}_p = [(\underline{c}^*_{p-1}-w_p J\underline{b}^*_{p-1})',w_p]$	$\underline{b}^{*\prime}_p = [(\underline{b}^*_{p-1}-k_p J\underline{b}^*_{p-1})',k_p]$

Table 1 - The General Levinson Algorithm recursions.



Auxiliary Problem
$\underline{b}^*_{p-1}$
k_{p-1}
$Q^*_{p-1}(\underline{r}_{p-1})$
(14)
(12)
(12)
(13)
(14)
(13)
$\underline{b}^*_p$
k_p
$Q^*_p(\underline{r}_p)$
$\underline{c}^*_{p-1}$
w_{p-1}
$Q^*_{p-1}(\underline{u}_{p-1})$
Original Problem
(17)
(16)
$\underline{c}^*_p$
(17)
w_p
(15)
(16)
(16)
$Q^*_p(\underline{u}_p)$

Fig. 1 - Flow diagram for Levinson recursions.

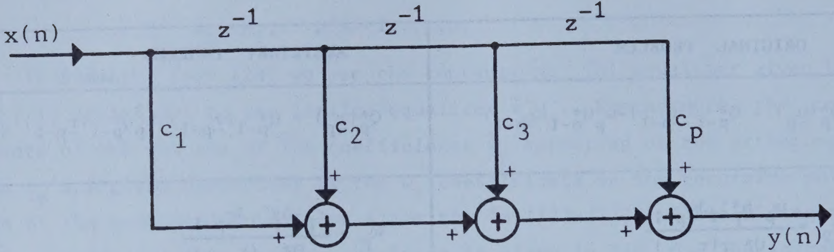

x(n)
z^{-1}
z^{-1}
z^{-1}
c_1
c_2
c_3
c_p
+
+
+
+
+
+
y(n)

Fig. 2 – TDL equalizer.

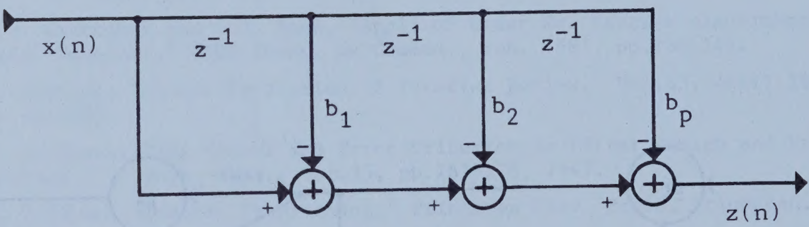

x(n)
z^{-1}
z^{-1}
z^{-1}
b_1
b_2
b_p
–
–
–
+
+
+
z(n)

Fig. 3 – TDL predictor

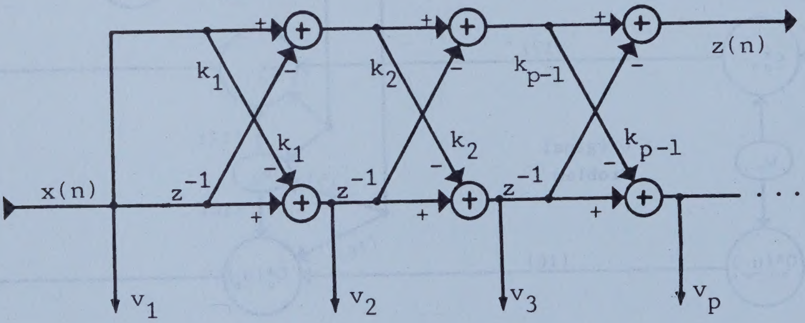

x(n)
z^{-1}
z^{-1}
z^{-1}
k_1
k_1
k_2
k_2
k_{p-1}
k_{p-1}
+
+
+
–
–
–
–
–
–
+
+
+
z(n)
v_1
v_2
v_3
v_p
. . .

Fig. 4 – Lattice predictor

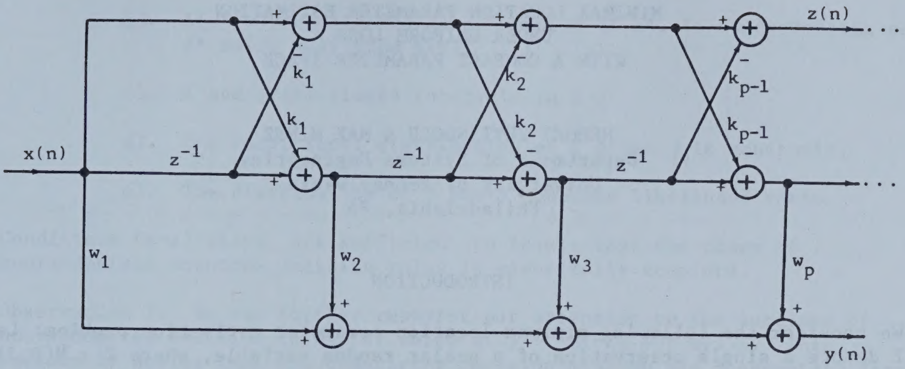

z(n)
x(n)
y(n)
z^{-1}
z^{-1}
z^{-1}
k_1
k_1
k_2
k_2
k_{p-1}
k_{p-1}
w_1
w_2
w_3
w_p
+
−

Fig. 5 - Lattice equalizer.

MINIMAX LOCATION PARAMETER ESTIMATION
UNDER UNIFORM LOSS
WITH A COMPACT PARAMETER SPACE

MEHMET ZEYTINOGLU & MAX MINTZ
Department of Systems Engineering
University of Pennsylvania
Philadelphia, PA

INTRODUCTION

We consider the following minimax location parameter estimation problem: Let Z denote a single observation of a scalar random variable, where $Z \in N(\theta, 1)$. Assume that θ is an unknown element of the given compact interval $\Omega = [-d, d]$. Let A denote the action space of the Statistician. Here, $A = [-d, d]$. Let $L(a,\theta)$ denote the uniform loss function defined on $A \times \Omega$:

$$\begin{aligned} L(a,\theta) &= 0, \qquad |a-\theta| \leq e; \\ L(a,\theta) &= 1, \qquad |a-\theta| > e; \end{aligned} \qquad (1)$$

where $e > 0$, is given. Based on these previous definitions and assumptions, we wish to determine a minimax estimate $\delta^*(Z)$ for θ. In this paper, we obtain minimax admissible Bayes estimates for θ. The minimax rules delineated in this paper exhibit several salient features:

a). The minimax rules are nonrandomized.

b). The prior distribution with respect to which a given δ^* is Bayes, is not necessarily a saddle-point (SP) strategy.

In viewing our present decision problem, in the context of the previous literature on minimax decision theory, we note that Wolfowitz (1950) solved the infinite interval version of the present problem. In particular, Wolfowitz showed that when $\Omega = E^1$, the equalizer rule $\delta^*(Z) = Z$, is minimax. A challenging aspect of this present problem is the compactness of Ω. Recent papers by Bickel (1981), and Casella and Strawderman (1981) have explored the implications of a compact interval parameter space for minimax estimation problems under quadratic loss. Earlier research on minimax estimation theory dealing with compact parameter spaces has appeared in papers by Nelson (1966) and Ghosh (1964). The "blind" version of our present problem, i.e., the limiting case where the Statistician makes no observation, has been considered by Egerland (1979). In this extreme case, the minimax strategy is a randomized procedure, which is characterized by a discrete distribution with a finite number of atoms in [-d, d].

PRELIMINARY OBSERVATIONS

Observation 1. We can restrict our attention to nonrandomized monotone decision rules. This follows by noting that the underlying decision problem is a monotone estimation problem in the sense of Karlin and Rubin (1956). (See also Brown (1976) and Berger (1980).) In particular, we note that:

a). $L(a^*,\theta)$ attains its minimum, as a function of a^*, at a point $a^* = q(\theta)$, where q is a nondecreasing function of θ;

b). $L(a^*,\theta)$, considered as a function of a^*, is nondecreasing as a^* moves away from $q(\theta)$;

c). $\mathcal{A}$ and Ω are closed intervals in E^1;

d). The conditional distribution of Z given θ is nonatomic;

e). The distribution of Z has a monotone likelihood ratio.

Conditions (a-e) above, are sufficient to insure that the class of nonrandomized monotone decision rules is essentially complete.

Observation 2. We can further restrict our attention to the subclass of nonrandomized decision rules for which $\delta(Z) = -\delta(-Z)$ for all Z. This follows by noting that the underlying decision problem has a symmetric loss function, and that the sampling density is symmetric about its mean.

Observation 3. The range space of the admissible decision rules is restricted to the interval $[-(d-e), (d-e)]$. This follows by noting that $L(d-e,\theta) = 0$, for $\theta \in [d-e,d]$, with a corresponding result for negative arguments.

We denote the conditional risk function of δ by $r(\delta,\theta)$.

Observation 4. In the present decision problem, $r(\delta,\theta)$ is given by:

$$r(\delta,\theta) = \Pr[\delta(Z)>\theta+e\,|\,\theta] + \Pr[\delta(Z)<\theta-e\,|\,\theta]. \tag{2}$$

If δ is onto $[-(d-e), (d-e)]$, then $r(\delta,\theta)$ can be expressed by:

$$r(\delta,\theta) = F(-\delta^{-1}(\theta+e)+\theta) + F(\delta^{-1}(\theta-e)-\theta), \tag{3}$$

where F is the CDF of the standard normal and δ^{-1} denotes a restricted inverse of δ. We define δ^{-1} for nonnegative arguments as follows: Let $K(t) = \{z\colon \delta(z) = t\}$, for $t \in [0,d-e]$. If $K(t)$ contains one point z, then $\delta^{-1}(t) = z$. If $K(t)$ contains more than one point, then $\delta^{-1}(t) = \sup K(t)$.

A MINIMAX DECISION RULE δ*, WHEN d IS AN INTEGER MULTIPLE OF e

In this section, we construct a minimax rule δ^*, for the case when d is an integer multiple of e. Let $d = (2n+1)e + c$, where $n = 1,2,\ldots,$ and the parameter c equals zero (e) if d is an odd (even) multiple of e. Our construction of δ^* is based on a sequence of observations and lemmas which we develop in this section. Due to the existing symmetry in this problem setting, function definitions will be stated for nonnegative arguments.

Let Δ denote the following parameterized family of decision rules δ:

$$\delta(Z) = \begin{cases} d-e, & c+a_n+2ne \le Z; \\ \cdot & \\ \cdot & \\ Z-a_2, & c+a_2+2e \le Z < c+a_2+4e; \\ 2e+c, & c+a_1+2e \le Z < c+a_2+2e; \\ z-a_1, & c+a_1 \le Z < c+a_1+2e; \\ c, & c \le Z < c+a_1; \\ Z, & 0 \le Z < c; \end{cases} \tag{4}$$

where: $0 \leq a_1 \leq a_2 \leq \cdots \leq a_n < \infty$. The selection of the parameters $a_1, a_2, \ldots, a_n$ will be described in the sequel.

Let Λ denote the following parameterized family of probability distributions λ, with density functions $f(\theta)$, defined on Ω:

$$f(\theta) = \sum_{i=1}^{n+1} k_i [U(\theta+(2i-1)e+c) - U(\theta-(2i-1)e-c)]; \tag{5}$$

where: $U(t) = 1$, if $t \geq 0$; $U(t) = 0$, if $t < 0$; and the parameters $\{k_i\}$, are nonnegative normalizing constants.

Lemma 1. Each $\delta \in \Delta$ (4) is Bayes with respect to an a priori density function, $f \in \Lambda$, for some choice of parameters $\{k_i\}$.
Proof: See Appendix 1.

Observation 5. The conditional risk function of each $\delta \in \Delta$ is given by:

$$r(\delta,\theta) = \begin{cases} F(a_n-e), & d-2e < \theta \leq d; \\ F(a_{n-1}-e), & \theta = d-2e; \\ F(-a_n-e)+F(a_{n-1}-e), & d-4e < \theta < d-2e; \\ \vdots & \\ F(-a_2-e)+F(a_1-e), & c+e < \theta < c+3e; \\ F(-a_2-e)+(c/e)F(-e)+(1-c/e)F(-a_1-e), & \theta = c+e; \\ F(-a_1-e)+(c/e)F(-e)+(1-c/e)F(-a_1-e), & 0 < \theta < c+e; \\ 2F(-a_1-e), & \theta = 0; \end{cases} \tag{6}$$

We observe that $r(\delta,\theta)$ is piecewise constant over the sets of a finite partition of Ω. We note specifically, that $r(\delta,\theta)$ is piecewise constant over n+1 nondegenerate subintervals of [0, d], where a nondegenerate interval is an interval which consists of more than a single point.

Based on this latter observation, we define an (n+1)-dimensional vector $J(\underline{a})$, with components equal to $r(\delta,\theta)$, over the n+1 nondegenerate partition intervals of [0, d].

$$\begin{aligned} J(\underline{a}) = [&(F(-a_1-e)+(c/e)F(-e)+(1-c/e)F(-a_1-e)), \\ &(F(-a_2-e)+F(a_1-e)), \\ &\vdots \\ &(F(a_n-e))]^T, \end{aligned} \tag{7}$$

where: $\underline{a} = (a_1, a_2, \ldots, a_n)^T \in \mathcal{D} \subset E^n$, such that $0 \leq a_1 \leq a_2 \leq \cdots \leq a_n < \infty$.

Observation 6. For each $\delta \in \Delta$, the max $r(\delta,\theta)$ occurs at one or more of the nondegenerate subintervals in (6).

Lemma 2. The parameter vector $\underline{a} \in \mathcal{D}$, can be chosen to equalize the

components of $J(\underline{a})$ to a common value M, where $M < 2F(-e)$.
Proof: See Appendix 2.

Lemma 3. If $\underline{a} \in \mathcal{D}$ satisfies the equalization condition of Lemma 2, then δ^* is minimax.
Proof: The proof follows by noting that δ^* is a Bayes rule with respect to a prior distribution λ^*, by Lemma 1; and that λ^* assigns probability one to the subset of Ω on which $r(\delta^*,\theta)$ attains its maximum value, by Lemma 2 and Observation 6.

Corollary 1. The prior distribution λ^* referred to in Lemma 3 is a least favorable prior distribution.
Proof: See Ferguson (1967) or Berger (1980).

AN EXAMPLE

We illustrate the previous results by means of the following example:

Let $d = 3e$, and $e = 0.1$. By referring to (4) and (6), the minimax rule δ^* and conditional risk function $r(\delta^*,\theta)$, are respectively:

$$\delta^*(Z) = \begin{cases} 2e, & a+2e \leq Z; \\ Z-a, & a \leq Z < a+2e; \\ 0, & 0 \leq Z < a; \end{cases}$$

$$r(\delta^*,\theta) = \begin{cases} F(a-e), & e < \theta \leq 3e; \\ F(-a-e), & \theta = e; \\ 2F(-a-e), & 0 \leq \theta < e; \end{cases}$$

for some $a > 0$.

Let $J(a) = (2F(-a-e),\ F(a-e))^T$. Upon equating the two components of $J(a)$, one obtains the relation $2F(-a-e) = F(a-e)$. Solving this last relation for a, we obtain $a^0 = 0.4$. The corresponding minimax risk is 0.62.

By employing equation (A.3) (from Appendix 1), we obtain the corresponding least favorable prior distribution λ^*, with density f^*:

$$f^*(\theta) = 0.14[U(\theta+e)-U(\theta-e)] + 1.62[U(\theta+3e)-U(\theta-3e)].$$

A MINIMAX DECISION RULE δ_M, WHEN d IS A NONINTEGER MULTIPLE OF e

In this section, we obtain a minimax decision rule δ_M, for the case when d is a noninteger multiple of e. We illustrate the main result by examining the special case when $d \in (e, 2e)$. The general case is addressed in an extended version of this paper. (See Zeytinoglu and Mintz (1982).)

We define a decision rule δ_M by:

$$\delta_M(Z) = \begin{cases} (d-e), & (d-e) \leq Z; \\ Z, & -(d-e) \leq Z \leq (d-e); \\ -(d-e), & Z \leq -(d-e). \end{cases}$$

Observation 7. The decision rule δ_M is Bayes with respect to the uniform distribution on $[-d, d]$. (See Appendix 1.)

Observation 8. The conditional risk function $r(\delta_M,\theta)$ is:

$$r(\delta_M,\theta) = \begin{cases} 0, & \theta \in [d-2e,\ 2e-d]; \\ F(-e), & \theta \in [-d,\ d-2e)\cup(2e-d,\ d]. \end{cases}$$

Observation 9. The decision rule δ_M is Bayes and unique up to equivalence. Thus, δ_M is admissible. (See Ferguson (1967).)

Lemma 4. The decision rule δ_M is a minimax rule.
Proof: If δ is any nonrandomized, odd, monotone (nondecreasing) decision rule, then either: $r(\delta,e^+) \geq F(-e)$, or $r(\delta,e^-) \geq F(-e)$; where e^+ and e^- denote respectively, the right-hand and left-hand limits of the indicated risk function. Thus, since $r(\delta_M,\theta) \leq F(-e)$ for all $\theta \in [-d,\ d]$, it follows immediately that δ_M is minimax.

Observation 10. The uniform distribution on $[-d,\ d]$ is not an SP strategy, since it does not assign probability one to the set of points on which $r(\delta_M,\theta)$ is maximized.

Thus, we have demonstrated the existence of a minimax admissible Bayes rule, where the prior distribution is not an SP strategy. This interesting result is a common characteristic of this class of minimax decision problems when d is not an integer multiple of e.

EXTENSIONS

In obtaining the results delineated in this paper, the CDF F( ) of the standard normal distribution has been utilized. However, all of the results which have been obtained in this paper remain valid when the normality assumption is replaced with a larger class of distributions with corresponding density functions that are unimodal, symmetric about θ, and possess a monotone likelihood ratio.

REFERENCES

Berger, J. O. (1980). Statistical Decision Theory. Springer-Verlag, New York.

Bickel, P. J. (1981). Minimax Estimation of the Mean of a Normal Distribution when the Parameter Space is Restricted. Ann. Statist. 9, 1301-1309.

Brown, L. D. (1976). Notes on Statistical Decision Theory. (Unpublished lecture notes, Ithaca.)

Cassella, G. and Strawderman, W. (1981). Estimating a Bounded Normal Mean. Ann. Statist. 9, 868-876.

Egerland, W. (1979). A Game-Theoretic Solution to an Engagement Problem. US Army Ballistic Research Laboratory Technical Report ARBRL-TR-02132. AD: A 066106.

Ferguson, T. S. (1967). Mathematical Statistics: A Decision-Theoretic Approach. Academic Press, New York.

Ghosh, M. N. (1964). Uniform Approximation of Minimax Point Estimates. Ann. Math. Statist. 35, 1031-1047.

Karlin, S. and Rubin, H. (1956). The Theory of Decision Procedures for Distributions with Monotone Likelihood Ratio. Ann. Math. Statist. 27, 272-299.

Nelson, W. (1966). Minimax Solution of Statistical Decision Problems by Iteration. Ann. Math. Statist. 37, 1643-1657.

Wolfowitz, J. (1950). Minimax Estimates of the Mean of a Normal Distribution with Known Variance. Ann. Math. Statist. 21, 218-230.

Zeytinoglu, M. and Mintz, M. (1982). Minimax Location Parameter Estimation Under Uniform Loss with a Compact Parameter Space. Submitted for publication.

APPENDIX 1

Lemma 1. Each $\delta \varepsilon \Delta$ (4) is Bayes with respect to an a priori density function $f \varepsilon \Lambda$ (5), for some choice of parameters $\{k_i\}$.
Proof: We restrict our attention to the specific case $d = 3e$, since it illustrates all the essential steps of a general proof.

Let λ denote a prior distribution with corresponding density:

$$f(\theta) = k_1[U(\theta+e)-U(\theta-e)] + k_2[U(\theta+3e)-U(\theta-3e)],$$

or equivalently,

$$f(\theta) = \begin{cases} (k_1+k_2), & \theta \ \varepsilon\ [-e,\ e] \\ k_2, & \theta \ \varepsilon\ [-3e,\ -e)\cup(e,\ 3e], \end{cases}$$

where, k_1 and k_2 are nonnegative normalizing constants.

Observation A1. The posterior density $f(\theta|Z)$, which corresponds to the posterior distribution $\lambda(\theta|Z)$ is:

$$f(\theta|Z) = (1/2\pi)^{\frac{1}{2}}G(Z)\exp[-(Z-\theta)^2/2]f(\theta), \tag{A.1}$$

where: $G(Z)$ is a normalizing factor.

Observation A2. A rule $\hat{\theta}(Z)$ is Bayes with respect to the prior distribution λ, if and only if it minimizes the posterior risk. Hence, for each Z, a Bayes rule must maximize:

$$A(\hat{\theta},Z) = \int_{\hat{\theta}-e}^{\hat{\theta}+e} f(\theta|Z)d\theta . \tag{A.2}$$

Observation A3. The maximization of $A(\hat{\theta},Z)$ over $\hat{\theta} \in [-2e, 2e]$ for fixed $Z > 0$, is carried out as follows:

First, evaluate $A(0,Z)$,

$$A(0,Z) = G(Z)(k_1+k_2)[F(e-Z)-F(-e-Z)] . \tag{A.3}$$

This latter expression corresponds to the case where $\hat{\theta}(Z) = 0$. Next, evaluate $dA(\hat{\theta},Z)/d\hat{\theta}$, when $0 < \hat{\theta} \leq 2e$. Through this analysis, we find that $A(\hat{\theta},Z)$ obtains a local maximum at:

$$\hat{\theta}^* = Z - a,$$

Where: $a = (1/2e)\ln[(k_1+k_2)/k_2]$.

The corresponding value of $A(\hat{\theta}^*,Z)$ is:

$$A(\hat{\theta}^*,Z) = G(Z)\{ k_1[F(e-Z)-F(-e-a)] + k_2[F(e-a)-F(-e-a)] \} . \tag{A.4}$$

Finally, by comparing (A.3) and (A.4), we obtain:

$$\begin{aligned} A(0,Z) &> A(\hat{\theta}^*,Z), & Z &\in [0,a); \\ A(0,Z) &= A(\hat{\theta}^*,Z), & Z &= a; \\ A(0,Z) &< A(\hat{\theta}^*,Z), & Z &> a. \end{aligned}$$

Observation A4. It follows from the previous observation that the rule:

$$\delta(Z) = \begin{cases} 0, & Z \in [0,\ a); \\ Z-a, & Z \in [a,\ a+2e); \\ 2e, & Z \geq a+2e; \end{cases}$$

where: $a = (1/2e)\ln[(k_1+k_2)/k_2]$;

is Bayes with respect to $f(\theta)$. We complete this observation with the remark that k_1 and k_2 can be chosen to simultaneously satisfy the previous equality and normalize $f(\theta)$.

APPENDIX 2

Lemma 2. A parameter vector $\underline{a} \in \mathcal{D}$, can be chosen to equalize the components of $J(\underline{a})$ to a common value M, where $M \in (0, 2F(-e))$.
Proof: We restrict our attention to the case where $c = 0$. The proof for the second case, $c = e$, follows in a similar fashion.

Observation A5. $F(X-e)+F(-X-e) > 2F(-e)$ for $X > 0$.

Observation A6. $F(X-e)$ $(F(-X-e))$ is an increasing (decreasing) function of X.

Step 1: Let $S_1(a_1) = 2F(-a_1-e)$, and $S_2(a_1) = F(a_1-e)+F(-a_2-e)$, where $a_2 \geq 0$ is fixed, and $a_1 \in [0, a_2]$. Consider the following sequence of observations:

- $S_1(a_1)$ is a decreasing function of a_1.
- $S_2(a_1)$ is an increasing function of a_1.
- $S_1(0) = 2F(-e)$, and $S_2(0) = F(-e)+F(-a_2-e)$.
- $S_1(0) > S_2(0)$.
- $S_1(a_2) = 2F(-a_2-e)$, and $S_2(a_2) = F(a_2-e)+F(-a_2-e)$.
- $S_1(a_1^0) = S_2(a_1^0) < 2F(-e)$, for $0 < a_1^0 < a_2$.
- $a_1^0 = g_1(a_2)$, where $g(t)$ is a continuous increasing function of t which passes through the origin.
- $S_1(a_1^0) = S_1(g_1(a_2))$ is a decreasing function of a_2.
- Since $S_1(g_1(a_2)) = S_2(g_1(a_2))$, $S_2(g_1(a_2))$ is a decreasing function of a_2.

Step 2: Let $T_1(a_2) = F(g_1(a_2)-e)+F(-a_2-e)$, and $T_2(a_2) = F(a_2-e)+F(-a_3-e)$, where $a_3 \geq 0$ is fixed, and $a_2 \in [0, a_3]$.

Proceeding as in Step 1, we determine $a_2^0 = g_2(a_3)$, such that $T_1(a_2^0) = T_2(a_2^0) < 2F(-e)$.

We continue in this fashion, choosing at each stage, say the i-th stage, the parameter a_i^0, as a function of the next parameter, i.e., $a_i^0 = g_i(a_{i+1})$. At the n-th stage, we have $Z_1(a_n) = F(-a_n-e)+F(g_{n-1}(a_n)-e)$, and $Z_2(a_n) = F(a_n-e)$. We observe that there exists an $a_n^0 \in (0,\infty)$, such that $Z_1(a_n^0) = Z_2(a_n^0) < 2F(-e)$. Once a_n^0 is thus determined, the other parameters $a_1^0, \ldots, a_{n-1}^0$ can be uniquely evaluated by using the previously established relations.

Thus, we have obtained a parameter vector $\underline{a}^0$ which equalizes the components of $J(\underline{a}^0)$ to a common value M, $M \in (0, 2F(-e))$.

ACKNOWLEDGMENT

The authors wish to express their gratitude for the following support. The research of the first author was supported in part by a Fulbright Grant. The research of the second author was supported in part by a National Research Council Associateship at the US Army Ballistic Research Laboratory.

SUFFICIENT STATISTICS FOR STATIONARY GAUSSIAN RANDOM PROCESSES

BRADLEY W. DICKINSON
Dept. of Electrical Engineering & Computer Science
Princeton University
Princeton, NJ 08544

ABSTRACT

It is known that the distribution of N samples of a stationary Gaussian autoregressive process admits a sufficient statistic whose dimension is independent of N. We will show that this property depends not on the absence of spectral zeros in autoregressive models, but rather on the fact that the class of models has a fixed set of spectral zeros.

RESULTS

Let $Y_N=(y_1, \cdots, y_N)$ be a set of observations from the stationary Gaussian, discrete-time random process $\{y_t, t\in\mathbf{Z}\}$. We assume that the process has zero mean and rational power spectral density. Equivalently, we assume that $\{y_t\}$ is an ARMA(p,q) process, satisfying the stochastic difference equation

$$y_t + a_1 y_{t-1} + \cdots + a_p y_{t-p} = u_t + b_1 u_{t-1} + \cdots + b_q u_{t-q} \tag{1}$$

where $\{u_t\}$ is an independent sequence of zero mean Gaussian random variables with $Eu_t^2=\sigma^2$. The parameters are denoted by $\vartheta=\{a_1, \cdots, a_p, b_1, \cdots, b_q, \sigma^2\}$.

We will give a characterization of subsets of the parameter space for which a sufficient statistic of fixed dimension, independent of the number of observation N, may be found. The autoregressive models, $q=0$, have this property (Arato, 1961), and we show that it is not the absence of spectral zeros that is important for this but rather the fact that the spectral zeros are fixed. That is, the moving average coefficients, $\{b_1, \cdots, b_q\}$ are fixed. This result is suggested by noting that the process $\{z_t\}$, defined by

$$z_t + b_1 z_{t-1} + \cdots + b_q z_{t-q} = y_t \tag{2}$$

is an autoregressive process derived from $\{y_t\}$ by filtering with a fixed, known filter. However, it is not possible to determine the sequence of observations $\{z_1, \cdots, z_N\}$ because this would require knowledge of the infinite past of $\{y_t\}$, so the autoregressive result cannot be used directly to give the result. An analysis of the structure of the density function for Y_N confirms the existence of a fixed dimensional sufficient statistic.

For observations from the process defined by (1), the probability density function for Y_N takes the form

$$p(Y_N) = c(\vartheta)\exp(-Y_N R_N^{-1} Y'_N/2) \ , \tag{3}$$

where $c(\vartheta)$ is a nomalizing constant, prime denotes transpose, and where R_N is the Toeplitz covariance matrix of Y_N. Denoting R_N^{-1} by $M_N=(m_{i,j}(\vartheta))$, we may write the natural logarithm of the density function in the exponential family form:

$$\log p(Y_N) = \log c(\vartheta) - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} m_{i,j}(\vartheta) y_i y_j \ . \tag{4}$$

If we can show that the dimension of the vector space spanned by the functions of $\vartheta, \{1; m_{i,j}(\vartheta), 1\le i,j\le N\}$ is a fixed number N_o, independent of N, we may choose a basis for this space , say $\{g_1(\vartheta), \cdots, g_{N_o}(\vartheta)\}$, and write

$$\log p(Y_N) = \sum_{j=1}^{N_o} g_j(\vartheta) h_j(Y_N) \ , \tag{5}$$

for suitable choices of the functions $\{h_j(Y_N)\}$. By the factorization theorem, these functions of the data may be taken as the components of a sufficient statistic whose dimension, N_o, is independent of N.

It therefore remains to study the form of the functions $m_{i,j}(\vartheta)$, obtained as elements of R_N^{-1}. We use a representation of R_N^{-1} due to Ljung amd Box (1979). Let $m = \max(p,q)$. For $N > m$,

$$R_N^{-1} = L'_A L'^{-1}_B (I_N - L_B^{-1} D W^{-1} D' L'^{-1}_B) L_B^{-1} L_A / \sigma^2 \ , \tag{6}$$

where L_A and L_B are the lower triangular Toeplitz matrices with first columns $(1, a_1, \cdots, a_p, 0, \cdots, 0)'$ and $(1, b_1, \cdots b_q, 0, \cdots, 0)'$, respectively, and D' is the m by N matrix $[\ I_m, O\ ]$. I with a subscript is the identity matrix of appropriate size. The m by m matrix W is given by

$$W = (V \Psi V')^{-1} + D' L'^{-1}_B L_B^{-1} D \ , \tag{7}$$

where $\sigma^2 \Psi$ is the $(p+q)$ by $(p+q)$ covariance of the vector of "unknown initial conditions" for (1). V is the m by $(p+q)$ matrix $[\ V_1\ V_2\ ]$, where V_1 is the m by p upper triangular Toeplitz matrix having first row $(-a_p, \cdots, -a_1)$ and V_2 is the m by q upper triangular Toeplitz matrix having first row $(b_q, \cdots, b_1)$.

The important point to notice is that only $L_A, V,$ and Ψ / σ^2 are nonconstant functions of the parameters $(a_1, \cdots, a_p)$. The elements of $(V \Psi V')^{-1}$ are rational functions of these variables, and the corresponding numerator and denominator polynomials, in reduced form, have terms whose degrees do not depend on N. Since D and L_B are independent of the parameters, the parameter dependence of W and W^{-1} is similar. Let $\delta(\vartheta)$ be the least common multiple of the denominator polynomials of the elements of W^{-1}. Then $W^{-1} = C(\vartheta) / \delta(\vartheta)$, where $\delta(\vartheta)$ and all of the elements of $C(\vartheta)$ are polynomials in $(a_1, \cdots, a_p)$ whose coefficients may depend on N but whose monomial terms have degrees that do not depend on N. From (6), we see that the elements of R_N^{-1} are also rational functions whose denominators may be taken to be $\delta(\vartheta)$ and whose numerators are linear combinations of products of the elements of $C(\vartheta)$ and quadratic polynomials in the variables $(a_1, \cdots, a_p)$. Therefore, the elements of R_N^{-1} are elements of a vector space consisting of rational functions having a common denominator polynomial and numerator polynomials of certain maximum degree, independent of N. Together with the function 1, these form a vector space whose dimension, N_o, is thus independent of N. This completes our demonstration that the ARMA(p,q) processes with fixed spectral zeros, i.e. fixed $(b_1, \cdots, b_q)$, admit a sufficient statistic whose dimension does not grow with the number of observations. It would be interesting to determine an explicit form for such a statistic.

ACKNOWLEDGMENT

This work was supported in part by the National Science Foundation, grant ECS82-05772. I am grateful to the Dept. of Electrical Engineering & Computer Science at the Johns Hopkins University for their hospitality during the Spring semester of 1982.

REFERENCES

M. Arato, "On the sufficient statistics for stationary Gaussian random processes," *Theory Prob. Appl.*, **6**, 199-201, 1961.

G.M. Ljung and G.E.P. Box, "The likelihood function of stationary autoregressive-moving average models," *Biometrika*, **66**, 265-270, 1979.

THE BOUNDED p-POINT CLASSES IN ROBUST HYPOTHESIS TESTING AND FILTERING*

SALEEM A. KASSAM
Department of Systems Engineering
Moore School of Electrical Engineering
University of Pennsylvania
Philadelphia, PA. 19104

ABSTRACT

The bounded p-point classes of density functions are defined through upper and lower bounds on the power or probabilities on different sets in some partition of the frequency or sample space. The robust Wiener filter is derived for signal and noise spectra lying in such classes, and is shown to be piecewise constant in its frequency response. Similar results are indicated for hypothesis testing involving such classes of probability densities. The robust matched filter is also obtained for the bounded p-point class of noise spectra, and is again shown to have a piecewise-constant frequency response.

I. INTRODUCTION

In recent years a considerable number of results has been obtained on minimax robust filters operating under conditions of inexact a priori knowledge of signal and noise characteristics. Specifically, filters for estimation, smoothing, prediction, and interpolation of random signals have been considered [e.g., 1-3], in addition to matched filters for signal-to-noise ratio optimization [e.g., 4], and other filters for special applications, e.g., time-delay estimation. In all these cases a priori uncertainties on signal and noise characteristics are modeled by classes of possible second-order characteristics, i.e., spectral densities or covariance functions. Many of these results were motivated by earlier work on robust hypothesis testing [5], in which minimax robust tests for binary hypotheses were obtained for some specific classes of probability density functions. In particular, minimax robust solutions for robust Wiener filtering under spectral uncertainty classes and for robust hypothesis testing under corresponding probability density function classes are very closely related.

Although some rather general results are available on robust hypothesis testing and robust Wiener filtering under general uncertainty classes, solutions for useful specific classes are of particular interest in applications. The spectral and probability density models for which specific solutions have been obtained include the following: 1) the ε-contaminated model [5]; 2) the "band-model" for bounded densities [6]; 3) the bounded total-variation model [5]; and 4) the specified p-point model [7].

Of these, perhaps of primary importance in applications are classes generated by the "band-models" and the specified p-point models. In the band-classes density functions are assumed to lie between upper and lower bounding functions, and in the specified p-point classes density functions are assumed to have specified proportions of their total area in specified regions in the sample-space or frequency domain. A natural and useful synthesis of these two types of classes yields the bounded p-point class of densities, where density functions are assumed to have proportions of their total area in specified regions lying between specified upper and lower bounds. It is easily seen that the bounded p-point classes include as

---

*This research is supported by the Air Force Office of Scientific Research under Grant AFOSR 82-0022.

a limiting case the band-classes, and as a special case the specified p-point classes.

We obtain the structure of robust Wiener filters for signal and noise spectra in bounded p-point classes defined by a common partition of the frequency space, by using general results on robust hypothesis testing for bounded classes [6]. The problem is first converted to one involving discrete spectra, and then results on robust hypothesis testing for discrete probability distributions are used. The robust Wiener filter has a piecewise constant frequency response. In a similar way it can be shown that robust hypothesis testing for bounded p-point classes of density functions requires the use of a quantizer operating on input data. For Wiener filtering we also obtain a result for the worst-case performance of any filter when signal and noise spectra lie in bounded p-point classes or the bounded classes of the "band" model. The above results are obtained for signal and noise classes defined on the same partition of the sample or frequency space. We also show that a solution can be obtained when one of the densities is in the bounded class generated by the band-model.

For matched filtering under noise spectral density uncertainty, we also consider the bounded p-point class for the noise spectra. For known signal the solution is simple. When the signal is also uncertain, lying in the previously used bounded L_2-deviation class [4], we can also obtain a general solution. The interesting result here is that the robust matched filter frequency response is again a piecewise-constant function.

II. ROBUST HYPOTHESIS TESTING AND WIENER FILTERING

We start by seeking a solution for the robust Wiener filter when signal and noise power spectral densities (PSD's) are members of bounded p-point classes. This will involve the robust binary hypothesis test for discrete "band" classes of probability densities. We define the bounded p-point class $\mathcal{S}$ of allowable signal PSD's S by

$$\mathcal{S} = \{S: \int_{A_j} S(f)df = \sigma_j^2 , \ j = 1,2,\dots,m, \ \sum_{j=1}^{m} \sigma_j^2 = \sigma^2\} \tag{1a}$$

Here $\{A_j\}_{j=1}^m$ is a given partition of the frequency space (with $f \in A_j \rightarrow -f \in A_j$), and σ^2 is the known total signal power, with the fractional powers σ_j^2 satisfying

$$\sigma_{Lj}^2 \leq \sigma_j^2 \leq \sigma_{Uj}^2 , \ j = 1,2,\dots m \tag{1b}$$

where the bounds σ_{Lj}^2, σ_{Uj}^2 are known. In a similar way we define the class $\mathcal{N}$ of allowable noise PSD's N by

$$\mathcal{N} = \{N: \int_{A_j} N(f)df = \nu_j^2, \ j = 1,2,\dots m, \ \sum_{j=1}^{m} \nu_j^2 = \nu^2\} \tag{2a}$$

and

$$\nu_{Lj}^2 \leq \nu_j^2 \leq \nu_{Uj}^2 , \ j = 1,2,\dots m \tag{2b}$$

Note that we are assuming here a common partition $\{A_j\}_{j=1}^m$ for both signal and noise classes.

We now consider a reduced discrete spectral distribution problem. Let all signal, noise power in each subset A_j be concentrated at a single frequency pair $\pm f_j \in A_j$. We will denote by $\mathcal{S}_d$, $\mathcal{N}_d$ the resulting classes of one-sided power spectral concentrations derived from $\mathcal{S}$, $\mathcal{N}$, respectively. Let H be the frequency response of any filter used for estimating the signal component with power spectrum $S_d = [S_d(1), S_d(2), \ldots S_d(m)]$ in $\mathcal{S}_d$, observed in additive uncorrelated noise with power spectrum $N_d \in \mathcal{N}_d$. Let $H_d(j) = H(f_j)$, $j = 1, 2, \ldots, m$. Then the mean-squared-error (mse) $e_d(S_d N_d; H_d)$ in estimating the signal for this discrete PSD's case is

$$e_d(S_d, N_d; H_d) = \sum_{j=1}^{m} S_d(j)\,|1-H_d(j)|^2 + N_d(j)\,|H_d(j)|^2 \tag{3}$$

For given S_d, N_d this is minimized for $H_d(j) = S_d(j)/[S_d(j)+N_d(j)]$ and the corresponding minimum mse is

$$e_d^*(S_d, N_d) = \sum_{j=1}^{m} S_d(j)N_d(j)/[S_d(j)+N_d(j)] \tag{4}$$

To obtain the least favorable pair (S_d^ℓ, N_d^ℓ), which is defined to be the pair maximizing (4), we examine the corresponding hypothesis testing problem for discrete densities. Consider a random variable with a finite set of possible realizations $\{1,2,\ldots,m\}$, with a probability mass function given by either $\{P_0(j)\}_{j=1}^m$ or $\{P_1(j)\}_{j=1}^m$. We assume that P_0 and P_1 are in the bounded classes defined by

$$\mathcal{P}_0 = \{P_0 : P_{0L}(j) \le P_0(j) \le P_{0U}(j)\ ,\ j = 1,\ldots,m\} \tag{5a}$$

$$\mathcal{P}_1 = \{P_1 : P_{1L}(j) \le P_1(j) \le P_{1U}(j)\ ,\ j = 1,2,..m\} \tag{5b}$$

Now the pair of densities (P_0^ℓ, P_1^ℓ) in these classes which is least-favorable (risk) in testing $\mathcal{P}_0$ vs. $\mathcal{P}_1$ [6] can be found by applying the general result in [6] for "band" classes of bounded probability density functions. The results in [6] are applicable for both absolutely continuous and discrete probability distributions. Note that the definition of the pair (P_0^ℓ, P_1^ℓ) which is least-favorable (risk) implies that a corresponding optimum test for P_0^ℓ vs. P_1^ℓ is robust for testing $\mathcal{P}_0$ vs. $\mathcal{P}_1$. On the other hand, the previous definition of the least-favorable pair (S_d^ℓ, N_d^ℓ) does not automatically imply robustness of the corresponding optimal filter, although for our case this will be shown to be true. In order to solve for the least-favorable pair (S_d^ℓ, N_d^ℓ) from the results in [6] applied to discrete probability distributions, we use the following lemma which is a direct counterpart of Lemma 1 in [2] where it is stated for the absolutely continuous case:

Lemma 1 Let $\mathcal{P}_0$ and $\mathcal{P}_1$ be classes of discrete probability mass functions $P = \{P(j)\}_{j=1}^m$ such that each member of $\mathcal{P}_0 \cup \mathcal{P}_1$ has positive values on the common set $\{1,2,\ldots m\}$. If P_0^ℓ and P_1^ℓ are least-favorable (risk) for $\mathcal{P}_0$ vs. $\mathcal{P}_1$ then

$$\sum_{j=1}^{m} \Psi[P_1^\ell(j)/P_0^\ell(j)]P_0^\ell(j) \ge \sum_{j=1}^{m} \Psi[P_1(j)/P_0(j)]P_0(j) \tag{6}$$

for all $P_0 \in \mathcal{P}_0$, $P_1 \in \mathcal{P}_1$ and all continuous concave functions Ψ.

The proof of this lemma follows the same ideas as those used in the proof of lemma 1 in [2]. Although this lemma can be extended to apply to the slightly more general case, we will instead restrict our original classes of spectral densities by requiring that the lower bounds σ^2_{Lj} and ν^2_{Lj} be positive (except in the case where both σ^2_{Uj} and ν^2_{Uj} are zero). Now we transform the original classes $\mathcal{S}_d$ and $\mathcal{N}_d$ into classes $\mathcal{P}_s$ and $\mathcal{P}_n$ of probability mass functions P_s and P_n respectively, by scaling each member of $\mathcal{S}_d$ by $1/\sigma^2$ and scaling each member of $\mathcal{N}_d$ by $1/\nu^2$. Thus, the $P_s \in \mathcal{P}_s$ are defined by $P_s(j) = S_d(j)/\sigma^2$, $j=1,2,\ldots,m$. Let $(P^\ell_s\ P^\ell_n)$ be the least-favorable (risk) pair in $\mathcal{P}_s \times \mathcal{P}_n$. Now e^*_d of (4) can be written as

$$e^*_d(P_s,P_n) = \sigma^2\nu^2 \sum_{j=1}^{m} \frac{L(j)}{\sigma^2 L(j)+\nu^2}\ P_n(j) \qquad (7)$$

where $L(j) = P_s(j)/P_n(j)$. The function $x/(\sigma^2x+\nu^2)$ is a concave function of x, and lemma 1 gives the result that $S^\ell_d = \sigma^2 P^\ell_s$ and $N^\ell_d = \nu^2 P^\ell_n$ form the least favorable pair maximizing (4) over the classes $\mathcal{S}_d$ and $\mathcal{N}_d$.

Once the least favorable pair (S^ℓ_d,N^ℓ_d) has been characterized, it remains to be shown that the optimum filter for this signal, noise spectra is the robust filter minimizing the maximum mse over the classes $\mathcal{S}_d$ and $\mathcal{N}_d$. This result is also a direct counterpart of Theorem 1 in [2] for the continuous case, and may be stated as follows:

Lemma 2 Let $\mathcal{S}_d$, $\mathcal{N}_d$ be convex classes of power spectral concentrations on a finite number of frequencies. Then $(S^\ell_d,N^\ell_d) \in \mathcal{S}_d \times \mathcal{N}_d$ is least-favorable for Wiener filtering iff (S^ℓ_d,N^ℓ_d) and $H_{d,R} = S^\ell_d/(S^\ell_d+N^\ell_d)$ form a saddlepoint for the mse functional $e_d(S_d,N_d;H_d)$.

Proof It is clear that if a pair (S^ℓ_d,N^ℓ_d) and its optimal filter $H_{d,R}$ form a saddlepoint for mse, then the pair is least-favorable. To prove the converse we note, as in the proof in [2], that the functional $e^*_d(S_d,N_d)$ is concave in S_d and in N_d, and that $\mathcal{S}_d$, $\mathcal{N}_d$ are assumed to be convex. From this the proof proceeds in a manner similar to that in [2], except that now we use the mse expressions (3) and (4) in place of the error expressions for the continuous case.

We thus find that the saddlepoint robust filter for the bounded classes $\mathcal{S}_d$, $\mathcal{N}_d$ of discrete spectra can be found as the optimal filter for the least-favorable pair in $\mathcal{S}_d \times \mathcal{N}_d$. This least favorable pair is obtainable as a scaled version of the least-favorable (risk) pair for testing the hypothesis $\mathcal{P}_s$ vs. $\mathcal{P}_n$ where $\mathcal{P}_s$ and $\mathcal{P}_n$ are derived by scaling members of $\mathcal{S}_d$ and $\mathcal{N}_d$ respectively. We now finally obtain the saddlepoint robust filter H_R for the original bounded p-point spectral classes $\mathcal{S}$ and $\mathcal{N}$.

Lemma 3 If $H_{d,R}$ is the saddlepoint robust filter for the classes $\mathcal{S}_d,\mathcal{N}_d$ of discrete spectra derived from the bounded p-point classes $\mathcal{S},\mathcal{N}$ defined by (1) and (2), then the filter defined by

$$H_R(f) = H_{d,R}(j),\ f \in A_j \qquad (8)$$

is the saddlepoint robust filter for spectral classes $\mathcal{S}$ and $\mathcal{N}$

Proof We have, for $S \in \mathcal{S}$ and $N \in \mathcal{N}$,

$$e(S,N;H_R) = \int S(f)\,|1-H_R(f)|^2 + N(f)\,|H_R(f)|^2 df \qquad (9)$$

$$= \sum_{j=1}^{m} |1-H_{d,R}(j)|^2 \int_{Aj} S(f)df + \sum_{j=1}^{m} |H_{d,R}(j)|^2 \int_{Aj} N(f)df$$

$$= \sum_{j=1}^{m} |1-H_{d,R}(j)|^2 S_d(j) + |H_{d,R}(j)|^2 N_d(j)$$

$$\leq \sum_{j=1}^{m} |1-H_{d,R}(j)|^2 S_d^{\ell}(j) + |H_{d,R}(j)|^2 N_d^{\ell}(j)$$

$$= e(S^{\ell}, N^{\ell}; H_R)$$

$$= e^{*}(S^{\ell}, N^{\ell}),$$

where S^{ℓ}, N^{ℓ} is any PSD's in $\mathcal{S},\mathcal{N}$ respectively for which $S^{\ell}(f)/N^{\ell}(f) = S_d^{\ell}(j)/N_d^{\ell}(j)$ when $f \in A_j$. Note that such a pair is least-favorable.

What we have obtained is an explicit solution for the piecewise-constant robust Wiener filter for bounded p-point classes of signal and noise spectra. In a similar way, we can obtain a corresponding result for robust hypothesis testing for bounded p-point classes of probability density functions.

Returning to the Wiener filtering problem, let $\mathcal{S}$ be a bounded p-point class of signal spectra, defined with respect to some partition $\{A_j\}_{j=1}^{m}$. Let $\tilde{\mathcal{N}}$ be any class of noise spectra, and let $\overline{\mathcal{N}}$ be the class of bounded p-point spectra generated by $\tilde{\mathcal{N}}$, for the same partition $\{A_j\}_{j=1}^{m}$. We then consider the least-favorable pairs $(S^{\ell},\overline{N}^{\ell})$ for these classes $\mathcal{S}$ and $\overline{\mathcal{N}}$ of signal and noise spectra. If a least-favorable pair exists with $\overline{N}^{\ell} \in \tilde{\mathcal{N}}$, then that pair is also least favorable for the original classes $\mathcal{S}$ and $\tilde{\mathcal{N}}$. We can apply this observation to the case where the class $\tilde{\mathcal{N}}$ is a "band" class of bounded spectra as in [1]. From the proof of lemma 3 we note that $S^{\ell},\overline{N}^{\ell}$ are required to be constant multiples of each other, their exact shapes not being constrained. Therefore it is always possible to find a least-favorable pair $(S^{\ell},\overline{N}^{\ell})$ for classes $\mathcal{S}$ and $\overline{\mathcal{N}}$ such that $\overline{N}^{\ell}$ is in the band class $\tilde{\mathcal{N}}$. This implies that the more precise bounds on noise spectra defining $\tilde{\mathcal{N}}$ represents extra information, beyond bounds on fractional powers, which is not useful in obtaining a better robust filter. Note that in the above, the roles of $\mathcal{S}$ and $\mathcal{N}$ can be interchanged and similar statements can be made. For hypothesis testing, again similar conclusions can be drawn.

Before considering matched filters in the next section, we establish a result which allows computation of the worst-case performance of any filter for signal and noise spectra defined either by "band" models or the bounded p-point classes.

In general, for given signal and noise classes it is of interest to have the worst-case performance of any given filter H. Consider the power-constrained band classes of spectra for Wiener filtering. Note that the mse obtained with any filter frequency response H is given by

$$e(S,N;H) = \int N(f)|H(f)|^2 df + \int S(f)|1-H(f)|^2 df$$

Consider any integral of the form $I = \int X(f)P(f)df$ where $P(f)$ is given and $X(f)$ lies in a power constrained "band" class of power spectra. Then the worst (maximum) value of I is obtained with X in the "band" class defined by

$$X_w(f) = \begin{cases} X_L(f), & P(f) < k \\ X_U(f), & P(f) > k \\ \text{arbitrary}, & P(f) = k \end{cases} \tag{10}$$

where k is a constant chosen to satisfy the power constraint. The proof follows easily by considering $\int[X(f)-X_w(f)]P(f)df$ for any other X. This result can be applied to each of the two terms in the above error expression to obtain the worst case signal and noise spectra in the "band" class for any given filter. A direct extension gives the worst-case spectra for the bounded p-point classes.

III. BOUNDED p-POINT NOISE CLASS IN ROBUST MATCHED FILTERING

For noise characteristics described in the frequency domain, the frequency response of the robust matched filter was derived explicitly in [4] for noise spectral uncertainty modeled by the "band" classes of allowable spectra. The signals were required to be within a given L_2 distance of a nominal signal. Here we will obtain results for the case where the noise class is the bounded p-point class described by (2). If S is the Fourier transform of the real deterministic signal and S_o is the nominal characteristic, the signal class $\mathcal{S}$ is now defined by

$$\mathcal{S} = \{S: \int_\Omega |S(f) - S_o(f)|^2 df \le \delta\} \tag{11}$$

Here Ω is $(-\infty,\infty)$ for the continuous case or $[-0{\cdot}5,\ 0{\cdot}5)$ for the discrete case. The signal-to-noise ratio obtained when using a filter frequency response H, when S and N are the given characteristics, is given by

$$\mathrm{SNR}(S,N;H) = \frac{|\int S(f)H(f)df|^2}{\int N(f)|H(f)|^2 df} \tag{12}$$

where integrals are over Ω. The following theorem gives our general result defining the least-favorable pair (S_R,N_R), which together with the corresponding optimum filter H_R forms a saddlepoint for the SNR functional.

Theorem 1 For the bounded p-point noise class of (2) and the bounded L_2-deviation signal class of (11), the pair (S_R,N_R) defined below is least-favorable for matched filtering and the optimum filter H_R for (S_R,N_R) is a saddlepoint robust (matched) filter, provided non-negative constants k and c exist satisfying (17) and (18). The pair (S_R,N_R) is defined by

$$S_{Rj}(f) = \begin{cases} [|S_o(f)| - k_j c]e^{i \arg S_o(f)}, & |S_o(f)| \ge k_j c \\ 0 & \text{, otherwise} \end{cases} \tag{13}$$

and

$$N_{Rj}(f) = |S_{Rj}(f)|/k_j, \tag{14}$$

for $j=1,2,\ldots,m$, where S_{Rj}, N_{Rj} are the restrictions of S_R, N_R to the sets A_j, respectively. Here k_j are defined as functions of S_R, N_R, k and c through the equations

$$\sigma_{Rj}^2 = \int_{A_j} |S_{Rj}(f)|df, \tag{15}$$

and

$$k_j = \begin{cases} \sigma_{Rj}^2/\nu_{Uj}^2 , & \sigma_{Rj}^2 > k\nu_{Uj}^2 \\ \sigma_{Rj}^2/\nu_{Lj}^2 , & \sigma_{Rj}^2 \le k\nu_{Lj}^2 \\ k , & k\nu_{Lj}^2 < \sigma_{Rj}^2 \le k\nu_{Uj}^2 \end{cases} \tag{16}$$

with the constraints

$$\sum_{j=1}^{m} \sigma_{Rj}^2/k_j = \nu^2 \tag{17}$$

and

$$\sum_{j=1}^{m} \{c^2k_j^2\int_{R_j} df + \int_{A_j-R_j} |S_o(f)|^2 df\} = \delta , \tag{18}$$

the set R_j, $j = 1,2,\ldots,m$, being defined as that subset of A_j on which $|S_o(f)|>k_jc$. In (14) for $N_{Rj}(f)$ we set $N_{Rj}(f)$ to be any function integrating to ν_{Lj}^2 on A_j if $\sigma_{Rj}^2 = 0$. The robust filter is

$$H_R(f) = \begin{cases} k_j \exp(-i\arg S_o(f)), & f \in R_j \\ S_o^*(f)/c & , f \in A_j-R_j . \end{cases} \tag{19}$$

Comments Note that (17) and (18) are the constraints on total noise power and on maximum deviation from the nominal signal for signals in S, respectively. The solution for S_R and N_R in any specific case requires a numerical procedure for evaluating the constants. In principle, this procedure can be an iterative procedure starting from some guess for and c. For given k and c a solution for the k_j and σ_{Rj}^2 can be sought from (15) and (16). Then the constraints (17) and (18) are tested. A proof is given in the Appendix.

It is also intersting to note that as the partition $\{A_j\}_{j=1}^m$ becomes finer the solution approaches that for the "band" class for N in [4]. In the special case where only a noise power constraint is given (j=1) the robust filter $H_R = S_R^*/N_R = k_j \exp(-i\arg S_o(f))$ on R_j has a constant magnitude.

From (13) and (14), the optimum filter for S_R, N_R is not uniquely defined where $|S_o|<k_jc$. On this set the robust filter is simply $S_o^*(f)/c$, i.e., a filter matched to S_o for white noise.

IV. ROBUST TIME-DELAY ESTIMATION

In a recent paper [8], the specified p-point classes and the band classes of bounded spectral densities have been used in obtaining robust versions of the Eckart filter [8], which is a specific type of weighting function used in the computation of the cross-correlation function in time-delay estimation. The basic problem is to estimate the time delay D between noisy signals received at two sensors, the sensor inputs being described by $v_1(t) = s(t) + n_1(t)$ and $v_2(t) = s(t-D) + n_2(t)$. Here $s(t)$ is a stationary random signal with spectrum S and n_1 and n_2 are uncorrelated noise processes independent of s, with spectra N_1 and N_2, respectively. The technique of time-delay estimation considered in [8] is based on forming an estimate of the cross-spectral density $G_{12}(f)$ of the inputs v_1 and v_2, weighting this with a filter function W, and then taking the inverse transform to obtain an estimate of the cross-correlation function $R_{12}(\tau)$. The position of the peak in $R_{12}(\tau)$ is taken as an estimate of the time delay D. For low signal-to-noise ratio at the input one appropriate performance measure is the output SNR of the estimate of $R_{12}(D)$. This is given by

$$d(S,Q;W) = \frac{[\int W(f)S(f)df]^2}{\int W^2(f)Q(f)df} \tag{20}$$

where $Q(f) = N_1(f)N_2(f)$. The spectral product Q and the signal spectrum can both be estimated from measurements at the system output. For given S and Q the optimum (Eckart) filter maximizing $d(S,Q;W)$ is $W_o = S/Q$, and this maximum distance is $d_o(S,Q) = \int S^2/Q\, df$.

The results in [8] show that for convex classes of allowable S and Q, the least-favorable pair (S_R, Q_R) for (S,Q), minimizing $d_o(S,Q)$, and the corresponding optimum filter $W_R = S_R/Q_R$ form a saddle point for $d(S,Q;W)$. Thus, the saddlepoint robust filter can be determined once the least favorable pair (S_R,Q_R) is known. Now since $d_o(S,Q)$ can be written as $\int (S/Q)^2 Qdf$, lemma 1 can be used to obtain directly the least favorable pair from the corresponding hypothesis testing problem. For S and Q lying in power constrained bounded p-point classes the least favorable pair is exactly the same as for Wiener filtering in Section II, where Q plays the role of a noise spectrum.

V. CONCLUSION

We have considered robust hypothesis testing and robust random signal estimation, deterministic signal detection and time-delay estimation, for cases where probability and spectral density functions belong to power-constrained bounded p-point classes. These classes are of practical interest because the bounds can be readily estimated in many situations. They form a generalization of the specified p-point classes and also of band-classes, which can be viewed as limiting cases of the bounded p-point classes.

REFERENCES

[1] S.A. Kassam and T.L. Lim, "Robust Weiner Filters," J. Franklin Inst., Vol. 304, pp. 171-185, 1977.

[2] H.V. Poor, "On Robust Weiner Filtering," IEEE Trans. Auto. Control, Vol. AC-25, pp. 531-536, 1980.

[3] K.S. Vastola and H.V. Poor, "Robust Wiener-Kolmogorov Theory," IEEE Trans. Inform. Theory, to be published.

[4] S.A. Kassam, T.L. Lim and L.J. Cimini, "Two-dimensional Filters for Signal Processing Under Modeling Uncertainties," IEEE Trans. Geoscience and Remote Sensing, Vol. GRS-18, pp. 331-336, 1980.

[5] P.J. Huber, "A Robust Version of the Probability Ratio Test," Ann. Math. Stat., Vol. 36, pp. 1753-1758, 1965.

[6] S.A. Kassam, "Robust Hypothesis Testing for Bounded Classes of Probability Densities," IEEE Trans. Inform. Theory, Vol. IT-27, pp. 242-247, 1981.

[7] L.J. Cimini and S.A. Kassam, "Robust and Quantized Wiener Filters for p-Point Spectral Classes," Proc. 14th Annual Conf. on Inform, Sciences and Systems, Princeton, pp. 314-319, 1980.

[8] E.K. Al-Hussaini and S.A. Kassam, "Robust Filters for Time-Delay Estimation Problems," Proc. 16th Annual Conf. on Inform. Sciences and Systems," Princeton, pp. 540-545, 1982

APPENDIX: Proof of Theorem in Section III.

Consider the denominator $D(N;H) = \int N|H|^2 df$ in (12). We have

$$D(N;H_R) - D(N_R;H_R) = \sum_j \int [k_j^2 R_j(f) + \frac{|S_o(f)|^2}{c^2} \bar{R}_j(f)]\,(N-N_R)df, \quad (A1)$$

where $R_j(f) = 1$ for $f \varepsilon R_j$ and 0 otherwise, and where $\bar{R}_j = A_j - R_j$. Now on $\bar{R}_j$ N_R is 0 and $|S_o|^2/c^2 \le k_j^2$, so

$$\begin{aligned} D(N;H_R)-D(N_R;H_R) &\le \sum_j k_j^2 \int [A_j(f)N - R_j(f)N_R]df \\ &\le \sum_{k_j>k} k^2[\int A_j(f)Ndf - \nu_{Uj}^2] \\ &+ \sum_{k_j<k} k^2[\int A_j(f)Ndf - \nu_{Lj}^2] + \sum_{k_j=k} k^2[\int A_j(f)Ndf - \sigma_{Rj}^2/k] \quad (A2) \end{aligned}$$

The RHS of (A2) is 0, so $D(N;H_R) \le D(N_R;H_R)$ for all $N \varepsilon \mathcal{N}$.

For the numerator, we first show that $\int SH_R df \ge 0$, all $S \varepsilon \mathcal{S}$. This follows by noting that, as in (A1),

$$H_R(f) = \sum_j [k_j R_j(f) + \bar{R}_j(f)|S_o(f)|/c]\exp(-i\arg S_o(f)),$$

replacing S by $S_o + (S-S_o)$. The integral containing S_o only is independent of S and is lower bounded by δ/c, from (18). The second integral [integrand containing $(S-S_o)$] is real-valued and, from Schwarz's inequality, bounded by δ/c in absolute value. Thus we consider minimization of this integral which is

$$\int [\sum_j k_j R_j(f) + \bar{R}_j(f)|S_o|/c]\,|S-S_o|\cos[\arg(S-S_o)-\arg S_o]df$$

To minimize the cos term is -1, when arg $(S-S_o)$ and arg S_o have a difference of π. Without the cos term, the integral is maximized for S given by (13), from a direct application of Schwarz's inequality and the above observation on phase.

OPTIMIZING THE MEASUREMENT PROCESSING ORDER FOR LINEAR REGRESSION WITH MULTIPLE INDEPENDENT VARIABLES

T. G. ROBERTAZZI
MANHATTAN COLLEGE
RIVERDALE, N. Y. 10471

S. C. SCHWARTZ
PRINCETON UNIVERSITY
PRINCETON, N. J. 08540

ABSTRACT

The problem of optimizing the measurement processing order for linear regression with multiple independent variables is considered. Two heuristic techniques for reducing the amount of computation necessary to produce such orderings are presented. One of them applies the accelerated greedy algorithm of M. Minoux to the ordering problem. The other involves restricting consideration, at each iteration, to locations adjacent to those already selected. An example which illustrates dramatic reductions in the number of incremental improvement evaluations that need to be computed is also included.

I. Introduction

Suppose that one is simultaneously presented with a "large" number of measurements, y_k, k=1, 2. . . . . M, taken at distinct and equi-spaced sampling locations in a multi-dimensional independent parameter space. It is assumed that the measurements are modeled as a linear combination of known basis functions in the presence of additive zero mean independent and identically distributed noise:

$$y_k = \sum_{i=1}^{N} \theta_i h_i(\underline{x}_k) + v_k \qquad k = 1, 2. \ldots . M \qquad (1.1)$$

Here the θ_i's are the parameters to be estimated, the h_i's are the corresponding basis functions and the $\underline{x}_k$'s are the multi-dimensional sampling locations.

Suppose also that the data is to be processed sequentially by one of the implementations of the recursive least squares algorithm. Then, an important question is whether there is some "optimal" order in which to process the measurements so that the most accurate estimates are obtained for the particular number of measurements already processed.

Why is this question of particular interest? Naturally all orderings produce equivalent statistical results once all M measurements have been processed. In addition, the off-line solution for the estimates based on M measurements is generally faster to compute then a sequential solution. However the sequential solution uses less storage and in a small processor it may be the only procedure available. The sequential algorithm also allows processing of just enough data to provide estimates of a desired statistical accuracy. This may be important if there is a cost associated with making each measurement. Finally the optimal ordering problem is closely related to the problem of finding the optimal subset of a given size from a set of equispaced measurements modeled as in (1.1).

In [1] this "optimal ordering problem" was examined for models in a single independent variable. It was shown that the problem can be formulated in terms of a vector valued optimization problem. A greedy (maximum immediate improvement) algorithm was presented for generating orderings. This algorithm can be viewed as a modified version of an algorithm which asymptotically produces optimal experimental designs. This close relationship not only provides a justification for the greedy ordering algorithm but also allows the nature of the orderings to be predicted.

In [2] an improved version of the greedy algorithm for producing experimental designs was presented. This was based on the accelerated greedy algorithm of M. Minoux [3]. In this paper it will be shown that this can also serve as the basis for an efficient (accelerated) higher-dimensional ordering algorithm. This is particularly useful for models with multiple independent variables.

This paper will also present an "adjacency restriction" heuristic to shorten the computation necessary to produce orderings. This merely consists of considering, at each iteration, only those measurement locations adjacent to locations which have already been selected. This simple heuristic can be used in conjunction with either the standard or the accelerated ordering algorithms and produces substantial savings with both procedures.

II. Greedy Algorithms

An optimal experimental design is a best allocation of a limited number of measurements. Since replicated measurements are generally allowed, this allocation involves both location and the fraction of the total number of measurements that will be made at each location. D-optimal designs seek to minimize the determinant of the covariance matrix associated with a particular design.

To understand the behavior of the greedy ordering algorithm, a review of the algorithm for asymptotically producing D-optimal designs is helpful. A large number of equi-spaced measurment locations is assumed to adequately cover the space of independent variables. For the N locations, the algorithm is:

Standard Greedy Design Algorithm

1. Choose an arbitrary initial allocation of replicated measurements.
2. Find: the location where an additional measurement appended to the current design produces the largest incremental improvement in the objective function (e. g. decrease in the error covariance matrix determinant). Add the measurement to the design. Continue (2) until some termination criteria is satisfied.

The first step of the algorithm is merely an initialization while the second step accounts for the "greedy" label. It can be demonstrated that this algorithm asymptotically produces D-optimal designs [4]. Actually, predicted measurement variance may be computed in place of incremental improvements in the determinant criteria [2,4].

To change the above procedure into a greedy ordering algorithm, each measurement location may be selected only once (replicated measurements are not allowed). The algorithm is initialized with the N locations corresponding most closely to a D-optimal design:

Standard Greedy Ordering Algorithm

1. Find the N optimal measurement locations, where N is the number of basis functions.
2. From the remaining M-N measurement locations, find the one which produces the largest incremental improvement in the objective function. Append this to the current intermediate ordering. Continue (2) until all the locations have been selected.

This algorithm can be seen to require approximately $M^2/2$ incremental improvement evaluations. This can be prohibitive as the number of independent variables increase More and more grid points are required to maintain the name accuracy in any individual independent variable. This is Bellman's "curse of dimensionality" [5]. For instance, if a two-dimensional rectangle consists of gxg measurement locations, then $g^4/2$ incremental improvement evaluations are necessary.

As an example, consider the following basis functions defined over a square sampling region ($x \in [-1, 1]$, $y \in [-1, 1]$):

$$1,\ x,\ y,\ xy,\ x^2,\ y^2 \tag{2.1}$$

In figure 1a through 1c a 31x31 grid has be used. The orderings generated after 9, 400, and 800 iterations are shown. It can be seen that the measurement locations along the border quickly form a single, connected region. The standard ordering algorithm requires almost half a million incremental improvement evaluations to produce this ordering (Table 1).

III. Accelerated Ordering Algorithm

The basic greedy algorithm involves, at each iteration, the calculation of a large number of incremental improvements and a search for the largest one. M. Minoux [3] recognized that in many problems, the incremental improvements can only decrease in magnitude from iteration to iteration. This property is called "sub-modularity". It allows a clever heuristic procedure to be used which avoids calculating many of the incremental improvements. Basically, incremental improvements are saved from iteration to iteration. These improvements are updated only until an updated one is found to be larger than all the others in storage. Since incremental improvements can only decrease in value, this must represent the maximal incremental improvement. The incremental improvement chosen to be updated is the largest one available in storage. Note that the point is that it is not necessary to update all the incremental improvements in order to find the maximal one.

This technique was used in [2] to produce an efficient design algorithm. With some minor modifications an efficient (accelerated) ordering algorithm can be produced.

Let E be the set of all measurement locations. Let S_i be the set of measurements associated with the ordering produced at the i^{th} iteration. Let e_j be the j^{th} measurement location. Let $d(S_i, e_j)$ be the incremental improvement that results when the e_j location is appended to the intermediate S_i ordering And let $f(S_i)$ be the value of the objective function (error covariance matrix determinant) for the S_i partial ordering.

Accelerated Ordering Algorithm

1. Begin with S_O initialized with the N optimal locations.
2. For every e_j belonging to E, compute:

$$d(S_O, e_j) = f(S_O + \{e_j\}) - f(S_O)$$

3. At iteration i, S_i is the current solution

   (a) Find:

$$e_j^{max} : \max_{e_j \in E-S_i} d(S_i, e_j)$$

   If i=O or if e_j has already been selected during this iteration, go to (3c).

   (b) Perform the update:

$$d(S_i, e_j^{max}) = f(S_i + \{e_j^{max}\}) - f(S_i)$$

   Then if

$$d(S_i, e_j^{max}) < \max_{e_j \in E-S_i} d(S_i, e_j)$$

   go to (3a). Otherwise go to (3c).

   (c) The maximal element is appended to the current set S_i:

$$S_{i+1} = S_i + \{e_j^{max}\}$$
$$i = i + 1$$

   If all the measurement locations have not been selected go to (3a).

The number of incremental improvement evaluations required by the accelerated design algorithm [2] is:

$$\sum_{i=1}^{\#iterations} c_{bf}(i)\, M \tag{3.1}$$

Here M is the number of measurement locations in the independent parameter space. The coefficients $c_{bf}(i)$ represent the fraction of locations whose associated incremental improvement is actually calculated during the i^{th} iteration. For the accelerated ordering algorithm the number of incremental improvement evaluations is:

$$\sum_{i=1}^{N} c_{bf}(i, M)\, M \tag{3.2}$$

However these c_{bf}'s are different from the coefficients of the design problem since the number of measurement locations is reduced by one after every iteration. That is, the coefficients depend on M as well as i.

For the particular polynomial example previously given, the accelerated ordering algorithm requires about 26,000 incremental improvement evaluations (Table 1) when using a 31x31 grid. This is a dramatic reduction over the standard algorithm's half a million evaluations.

IV. Adjacency Restriction

During the initial part of the orderings generated by the greedy ordering algorithm, measurements tend to be clustered about the D-optimal design points of support. This is a consequence of the close relationship between the greedy ordering algorithm and the greedy design algorithm. During the latter part of the orderings, measurement locations, at least, tend to be selected which are adjacent to other previously selected locations. This follows in part from the differentiable nature of commonly used basis functions.

It is natural then to consider restricting the calculation of incremental improvements to only those measurement locations which are adjacent to previously selected measurement locations. This should lead to a marked decrease in the number of evaluations since this heuristic avoids performing evaluations for every measurement location during such iteration. The ordering generated with the adjacency restriction will be significantly different from the nominal ordering only if there is some D-optimal design with measurement locations which are not present in the (different) D-optimal design that is used to initialize the ordering algorithm.

If the adjacency restriction is applied to the standard ordering algorithm, the locations selected by any particular iteration will generally form connected regions about the locations of the original design. Thus the evaluations will be performed along the boundaries of these n dimensional regions. As the number of locations in the overall sampling region, M, is increased, the number of iterations performed during any iteration will grow as the $(n-1)/n$ root of M. This means that over M iterations, the growth in the total number of incremental improvement evaluations required is:

$$M^{1+(n-1)/n} \tag{4.1}$$

If the adjacency restriction is applied to the accelerated ordering algorithm, similar reasoning leads to a total number of function evaluations of:

$$\sum_{i=1}^{M} C_{bf}(i, M)\, M^{(n-1)/n} \tag{4.2}$$

V. Heap Type Data Structure

In [2] the importance of using a heap type data structure for storing the incremental improvements was pointed out. This is necessary to avoid requiring an excessive amount of computation for the operation of finding the largest incremental improvement at each iteration. This structure can be readily adapted to the accelerated ordering algorithm. When a measurement location has been selected, its associated incremental improvement (actually a predicted measurement variance is stored in the heap, see [2]) is set to zero. This causes that location to drop to the bottom of the heap and not be selected in the future.

Incorporating the adjacency restriction into the heap structure is also simple. During any iteration, only those measurement locations which are adjacent to already selected locations will be present in the heap. If the heap is of fixed size, M, this means that most of the nodes of the heap will be empty (have the value zero). Naturally a procedure for incorporating newly adjacent locations must be incorporated into the algorithm.

VI. An Example

Figure 2 is a plot of minus the $\log_{10}$ of the error covariance matrix determinant for the four ordering algorithms which have been mentioned. It can be seen that the curves coincide and the performance of these algorithms is essentially identical.

Figure 3 is a plot of the number of incremental improvement evaluations, as a function of iteration, performed by the standard algorithm with the adjacency restriction, and also by the accelerated algorithm. The number of evaluations per iteration is largely constant for the former algorithm. Examining figure 1, it can be seen that the evaluations occur along two elliptical looking boundaries. As the outer one decreases in size, the interior one increases. This produces a relatively constant number of incremental improvement evaluations.

Figure 4 is a plot of, again, the number of incremental improvement evaluations for the standard algorithm with the adjacency restriction and also for the accelerated algorithm with the adjacency restriction.

Table 1 summarizes simulations made with equi-spaced grids of 961, 1681 and 2401 (measurement locations). In this table, the number of iterations is equal to the number of measurement locations, less nine (the number of locations in the original design).

The execution times required by the four algorithms appear in Table 2. Because an accelerated algorithm iteration is more time consuming than a standard algorithm iteration, the improvement is not as dramatic as in Table 1. However, had the basis functions used in the example been more complicated, a larger improvement in time would have been apparent.

From (4.2), there is a proportionality for the computation required by the accelerated ordering algorithm with the adjacency restriction to $M^{1/2}$. Tables 3 and 4 were constructed to illustrate this. The number of iterations is held constant for both tables while the number of measurement locations is varied. The first table lists the results for 961 iterations and the second table for 1681 iterations.

VII. Conclusion

Two heuristic techniques for the efficient computation of nearly optimal processing orders for linear regression have been presented. Both techniques can be exploited simultaneously in the form of the accelerated ordering algorithm with adjacency restriction. This new algorithm appears to provide an efficient way of generating orderings, particularly in the case of multiple independent variables.

References

1. T.G. Robertazzi and S.C. Schwartz, "Optimizing the Measurement Processing Order for Linear Regression", Proc. of the Fifteenth Annual Conf. on Information Sciences and Systems, John Hopkins University, Baltimore, Maryland, March 1981, pp. 507-511.
2. T.G. Robertazzi and S.C. Schwartz, "An Accelerated Design Algorithm", 10th IFIP Conference on System Modeling and Optimization, Springer-Verlag, New York, September 1981,
3. M. Minoux, "Accelerated Greedy Algorithms for Maximizing Submodular Set Functions", 8th IFIP Conference, Wurzburg, 1977, Springer-Verlag, Part 2, pp. 234-243.
4. V.V. Federov, Theory of Optimal Experiments, Academic Press, 1972.
5. L. Cooper and D.I. Steinberg, Introduction to Methods of Optimization, Prentice-Hall, 1963, pg. 17.

Acknowledgement:

This research was supported in part by the Office of Naval Research under contract N00014-81-K-0146.

First 9 Measurement Locations

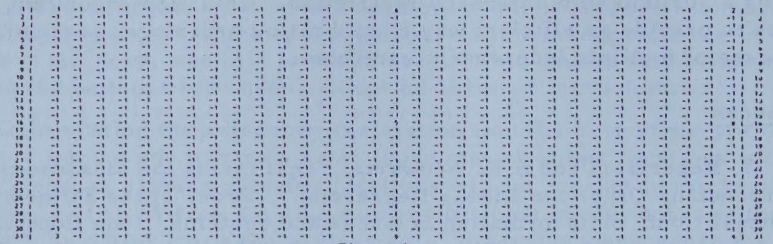


Figure 1a

First 400 Measurement Locations

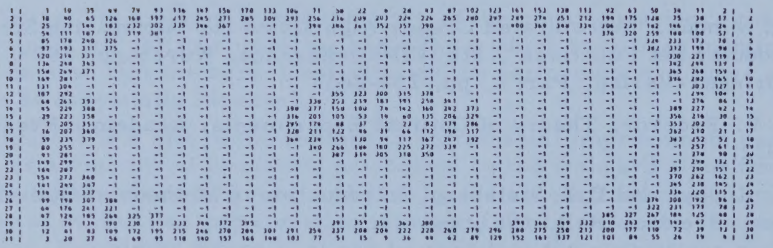


Figure 1b

First 800 Measurement Locations

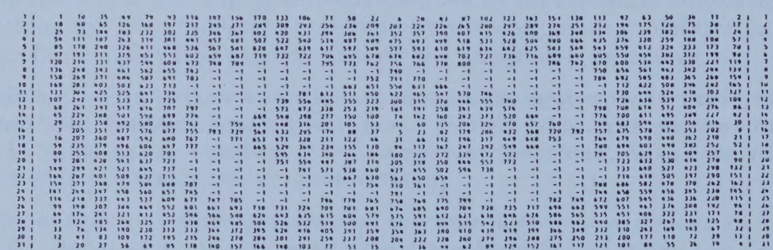


Figure 1c

Table 1

Number of Function Evaluations			
M	961	1681	2401
Standard	462,241	1,413,721	2,883,601
Accelerated	26,416	55,362	87,792
Stand. with Restr.	119,317	278,543	478,649
Accel. with Restr.	14,601	27,956	42,168
Linear Increase in M		1.75	1.43
Actual Incr., Accel.		2.1	1.6
Power 3/2 Incr. in M		2.3	1.7
Act. Incr., Stand. w. Restr.		2.3	1.7
Act. Incr., Accel. w. Restr.		1.9	1.5

Table 2

Execution Time (Seconds)			
M	961	1681	2401
Standard	26.	78.	157.
Accelerated	10.	23	38.
Stand. with Restr.	23.	54.	93.
Accel. with Restr.	7.	15.	25.
Quad. Incr. in M		3.0	2.0
Act. Incr., Stand.		3.0	2.0
Power 3/2 Incr. in M		2.3	1.7
Act. Incr., Stand. w. Restr.		2.3	1.7

Table 3

Number of Function Evaluations		
	M = 961	M = 1681
961 iterations	14,601	19,285
Power 1/2 Increase in M		1.3
Actual Increase in M		1.3

Table 4

Number of Function Evaluations		
	M = 1681	M = 2401
1681 iterations	27,956	30,817
Power 1/2 Increase in M		1.2
Actual Increase in M		1.1

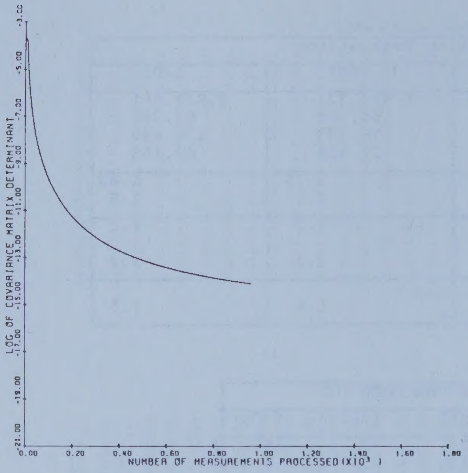


Figure 2

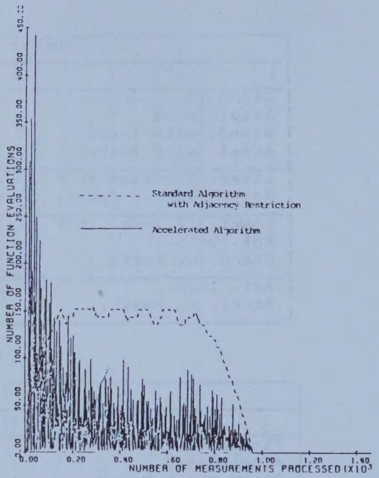


Figure 3

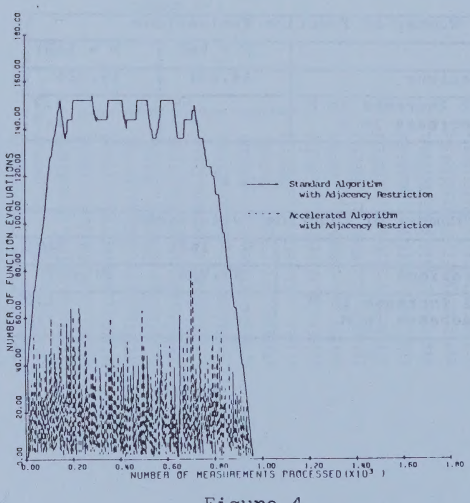


Figure 4

DETECTION AND ESTIMATION WITH FIXED LAG FOR ABRUPTLY CHANGING SYSTEMS

V. JOHN MATHEWS
University of Iowa, Iowa City, Iowa

JITENDRA K. TUGNAIT[†]
Exxon Production Research Company, Houston, Texas

ABSTRACT

The problem of state estimation and system structure detection for discrete-time stochastic systems with parameters which may switch among a finite set of values is considered. The switchings are modelled by a semi-Markov, or Markov, chain with known transition statistics. A fixed time-delay (lag) is allowed in estimation (smoothing) and detection. The optimal solutions require geometrically increasing computations and storage with time. Suboptimal solutions are proposed to alleviate this problem and simulation results are presented to illustrate the effectiveness of the proposed algorithms and the advantages of introducing a delay in processing of the observations.

I. INTRODUCTION

This paper is concerned with state estimation and system structure detection for linear discrete-time stochastic systems with abruptly changing parameters. It is assumed that a fixed time-delay (lag) is allowed in estimation and detection, i.e., the processes of estimation and detection at time t utilize measurements till time t+N where the lag N is a nonnegative integer. The abruptly changing parameters are modelled as a finite-state semi-Markov, or Markov, chain with known transition statistics.

Motivation for considering system models with jumps stems from applicability of such models to a large class of realistic problems. Failures in components or subsystems of a dynamical system can be represented by abrupt changes in the system parameters [13]-[15]. Similarly, repairs of the failed components and system reconfigurations also cause abrupt changes in the system parameters [14]. Approximation of nonlinear systems by a set of linearized models to cover the entire dynamic range can also lead to linear systems with jump parameters [16]-[18]. Modelling with semi-Markov or Markov jump parameters may also be thought of as an extension of the multiple model partitioning approach to estimation, detection, and control of systems with unknown time-invariant parameters [19]-[22], to include systems with (abruptly) time-varying parameters [4], [7]. For some other motivations, see [1], [3], and [23].

---

[*] The work of V.J. Mathews was supported by the University of Iowa under a Teaching and Research Fellowship.

[†] The work of J.K. Tugnait was supported by National Science Foundation under Grant ECS-80-05956 at the Dept. of Electrical and Computer Engineering, University of Iowa, Iowa City, Iowa.

The problem of state estimation and systems structure detection with zero lag, for abruptly changing systems, has received considerable attention [1]-[7], [16]. A recent survey concerning lienar discrete-time systems with Markov jump parameters may be found in [7]. In [1], [2], [4]-[7], the abruptly changing parameters have been modelled as a finite-state Markov chain whereas, in [2], [16]-[18], [23], and [25], a semi-Markov chain model has been employed. The semi-Markov chain model is more general in that it includes the Markov chain model [9]-[11]. The optimal solutions to the filtering and the detection problems are intractable, therefore, in [1]-[7] and [16], efforts are directed toward finding suboptimal solutions.

For linear systems with completely known parameters, it is well-known that state estimation with fixed time-delay (fixed-lag smoothing) leads to an improvement in the performance at the cost of increased complexity when compared with the zero-lag case [8], [12]. In this paper we examine the consequences of introducing a fixed lag for systems with jump parameters. It is assumed that this time-delay is of little consequence in the intended application. This is certainly true if the intended application is off-line analysis of a maneuvering target tracking problem [3], [4], [16].

II. PROBLEM STATEMENT

Let $r(t) \varepsilon S = \{1, 2, \ldots, s\}$, $t \varepsilon \{0, 1, 2, \ldots\}$, denote a finite-state, discrete-time, stochastic process (either a Markov chain or a semi-Markov chain) with completely known probability laws. The process $r(t)$ governs the structure of a stochastic dynamical system. The system state equation is given by

$$x(t+1) = A(r(t+1))x(t) + B(r(t+1))w(t) + b(r(t+1)) \qquad (1)$$

where $x(t) \varepsilon R^n$ is the system state, $w(t) \varepsilon R^{\ell}$ is a zero-mean white Gaussian noise sequence with covariance Q, and $b(r(t+1)) \varepsilon R^n$ is a "bias" input which cannot be measured (e.g., it may be inaccessible). The matrices $A(r(t))$ and $B(r(t))$ are functions of the chain $r(t)$ and so is the vector $b(r(t))$. The observation equation associated with (1) is modelled by

$$z(t) = C(r(t))x(t) + D(r(t))v(t) + g(r(t)) \qquad (2)$$

where $z \varepsilon R^m$ is the observation vector, $v \varepsilon R^m$ is a zero-mean white Gaussian measurement noise sequence with covariance R such that $D_i R D_i^T > 0$[†] $(\forall i \varepsilon S)$ where $D(r(t)) \varepsilon \{D_i,\ i = 1, 2, \ldots, s\}$, and $g(r(t))$ is an unmeasurable "bias" input. The initial state $x(0)$ is assumed to be Gaussian with mean $\bar{x}_0$ and covariance matrix P_0. Finally, $x(0)$, $v(t)$, $w(t)$, and $r(t)$ are mutually independent.

The objective is to find the MMSE (minimum mean-square error) smoothed estimate $\hat{x}(t-N|t)$ of system state $x(t-N)$ given the observations $Z_t \triangleq \{z(k),\ 0 \leq k \leq t\}$, and to decide on the value of $r(t-N)$ (system structure detection), given Z_t, minimizing the probability of error. Here N (a fixed positive integer) is the fixed lag in estimation and detection.

[†] $R>0$ denotes that the matrix R is positive-definite.

III. OPTIMAL SOLUTION

IIIA. Fixed-Lag Smoothing

It is well known that the MMSE smoothed state estimate $\hat{x}(t-N|t)$ is given by the conditional mean

$$\hat{x}(t-N|t) = E\{x(t-N)|Z_t\} \tag{3}$$

Define a system structural state sequence $I(t)$ as

$$I(t) = \{r(0), r(1), \ldots, r(t)\} \tag{4}$$

Denote the smoothed state estimate conditional on a specific sequence as

$$\hat{x}_j(t-N|t) \overset{\Delta}{=} E\{x(t-N|Z_t, I_j(t)\} \tag{5}$$

where $I_j(t)$ denotes a specific sequence from the space of all possible sequences $I(t)$. This leads to

$$\hat{x}(t-N|t) = \sum_{j=1}^{s^{t+1}} \hat{x}_j(t-N|t)P(I_j(t)|Z_t) \tag{6}$$

where $P(I_j(t)|Z_t)$ is the conditional probability of the sequence $I_j(t)$ given the observations Z_t. By the Bayes' rule it follows that

$$P(I_j(t+1)|Z_{t+1}) = \frac{f(z_{t+1}|Z_t,I_j(t+1))P(I_j(t+1)|Z_t)}{\sum_{n=1}^{s^{t+2}} f(z_{t+1}|Z_t,I_n(t+1))P(I_n(t+1)|Z_t)} \tag{7}$$

where $f(z_{t+1}|Z_t,I_j(t+1))$ is the conditional probability density of the ovservation Z_{t+1} given the past observations Z_t and the particular structural state sequence $I_j(t+1)$. Furthermore, we have

$$P(I_j(t+1)|Z_t) = P(I_m(t)|Z_t)P(r(t+1)|Z_t,I_m(t)) \tag{8}$$

where $I_j(t+1) = \{I_m(t),r(t+1)\}$ with $r(t+1)\varepsilon S$, i.e., $I_m(t)$ is a subsequence of $I_j(t+1)$ and $r(t+1)$ is the "last" or "most recent" element of $I_j(t+1)$. Computation of the second quantity of the right side of (8) depends upon the nature of the probability law of $\{r(t)\}$ and will be be discussed later in the section.

Now $f(z(t+1)|Z_t,I_j(t+1))$ and $\hat{x}_j(t-N|t)$ can be calculated recursively by applying Kalman filtering/smoothing methods to the system model (1)-(2). The initial condition $x(0)$ has a Gaussian distribution. Given the sequence $I_j(t+1)$, the system (1)-(2) is now a linear system with Gaussian noise and known parameters. Therefore, equations for $\hat{x}_j(t-N|t)$ are those for a Kalman smoother [8, Sec. 7.3] matched to the sequence $I_j(t+1)$. Finally, it is easy to show that (see, e.g., [1]) $f(z(t+1)|Z_t,I_j(t+1))$ is Gaussian whose mean and covariance can be calculated by using the various quantities associated with the Kalman filter/smoother matched to the sequence $I_j(t+1)$.

It remains to calculate $P(r(t+1)|Z_t,I_m(t))$ needed in (8). This depends upon the nature of the process $r(t)$. We discuss two cases.

IIIA.1 Markov Jump Parameters

In this case the probability law for the Markov chain $r(t)$ is completely specified [11] by specifying the initial probabilities $P\{r(0) = 1\}$, $i\varepsilon S$, and the transition probabilities $p_{ij} = P\{r(t+1) = j \mid r(t) = i\}$, $(i,j\varepsilon S)$; for simplicity, we assume that the Markov chain is homogeneous, i.e., p_{ij} is not a function of time t. It is assumed that these probabilities are known. With $r(t)$ modelled as a Markov chain, we have

$$P(r(t+1) \mid Z_t, I_m(t)) = P(r(t+1) \mid I_m(t)) = P(r(t+1) \mid r(t) = i) \tag{9}$$

where the first step in (9) follows from independence of $r(t+1)$ and Z_t given $I(t)$ and the second step in (9) follows from the Markovian nature of $r(t)$ [11]. In (9), $r(t) = i$ is the "last" element of sequence $I_m(t)$.

IIIA.2 Semi-Markov Jump Parameters

Suppose the process $\{r(t),\ t \geq 0\}$ is modelled as a homogeneous semi-Markov chain [9]-[11]. The semi-Markov process is a renewal process that makes its transitions according to the transition probability matrix of a Markov process, but whose time between transitions can be an arbitrary random variable. The Markov chains are special cases of the semi-Markov chains. By independence of $r(t+1)$ and Z_t given $I(t)$, it follows that

$$P(r(t+1) \mid Z_t, I_m(t)) = P(r(t+1) \mid I_m(t)) \tag{10}$$

To compute the above quantity, we need to define the following probabilities. With $\ell\varepsilon S$, let $(n \geq 0)$

$$P_\ell(n) \overset{\Delta}{=} \Pr\{r(t) \neq r(t-1) \mid r(t-1) = r(t-2) = \dots = r(t-n+1) = \ell \neq r(t-n)\} \tag{11}$$

i.e., $P_\ell(n)$ is the probability that a jump (transition) occurs at time t given that the last jump occurred at time $t-n+1$ to state $\ell\varepsilon S$. By homogeneity of the semi-Markov chain, $P_\ell(n)$ is not a function of time t. Also, we must have $\sum_{n=0}^{\infty} P_\ell(n) = 1$ for each $\ell\varepsilon S$. It should be noted that $P_\ell(n)$ defined in (11) is a function of n ("holding" or "sojourn" time) and ℓ (the "current" state) only. Furthermore, we have [9]-[11]

$$\beta_{\ell i}(n) = P_\ell(n) p_{\ell i} \tag{12}$$

where $\beta_{\ell i}(n)$ denotes the probability that a jump occurs to state i at time given that the last jump occurred to state ℓ at time $t-n+1$, and $p_{\ell i}$ is the transition probability of the imbedded Markov chain as defined in Section IIIA.1. It is assumed that $P_\ell(n)$ $(\forall \ell\varepsilon S$ and $\forall n \geq 0)$ and p_{ij} $(\forall i,j\varepsilon S)$ are completely known.

To rewrite (1) in terms of $P_\ell(n)$ and p_{ij}, let $I_m(t) = \{r(0), r(1), \dots, r(t)\colon\ r(t) = r(t-1) = \dots = r(t-j+1) = \ell \neq r(t-j),\ j \geq 0\}$, i.e., $I_m(t)$ is such that the last transition occurs at time $t-j+1$ to state $\ell\varepsilon S$. Then we have

$$P(r(t+1) \mid I_m(t)) = \beta_{\ell i}(j+1) \text{ if } r(t+1) = i \neq \ell \text{ and } = 1-P_\ell(j+1) \text{ if } r(t+1) = \ell \tag{13}$$

It is easy to see from (6)-(10) and (13), that $\hat{x}(t-N|t)$ can be computed recursively by running s^{t+1} "elemental" smoothers in parallel. For large t, this is clearly impractical. Therefore, one has to resort to suboptimal schemes. In the next section we propose a suboptimal algorithm based on the detection-estimation approach of [6], [7].

IIIB. Structure Detection

Find $\hat{r}(t-N)$ such that

$$\hat{r}(t-N) = \arg\left\{\min_{i\varepsilon S} P(i \neq r(t-N)|Z_t)\right\} . \tag{14}$$

It is well known [12] that in such case we must have

$$\hat{r}(t-N) = \arg\left\{\max_{i\varepsilon S} P(r(t-N) = i|Z_t)\right\} . \tag{15}$$

Now we have

$$P(r(t-N) = i|Z_t) = \sum_{j=1}^{s^t} P(I_j(t)|Z_t) \tag{16}$$

where $I_j(t)$ is a specific sequence from the space of all sequences I(t) as defined in (4) such that its (t-N+1)st element $r(t-N) = i$. Note that out of s^{t+1} possible sequences at time t, s^t sequences satisfy this criterion.

Now (16) ($\forall i\varepsilon S$) can be computed recursively using (7)-(10) and (13), by running s^{t+1} elemental smoothers in parallel, as in Section IIIA. The approximation used to alleviate this difficulty is the same as for the smoothing problem; we discuss it is the next section.

IV. SUBOPTIMAL SOLUTION

In this section a suboptimal solution to the optimal smoothing problem is proposed. It is an extension of the detection-estimation approach to filtering proposed in [6]. The same approximation is used for structure detection also. The scheme consists of running at most M (instead of s^{t+1}) elemental smoothers in parallel at any time t where M is a fixed positive integer dictated by the computation and storage capabilities of the processor. A maximum of M "most likely" structural state sequences are selected based on their a posteriori probabilities $P(I(t)|Z_t)$ and then the state estimate is based on the selected sequences only.

Suppose that, at time t-1, we have selected L_{t-1} system structural state sequences where $L_{t-1} \leq M$. At time t consider all possible extensions of these L_{t-1} state sequences. By an extension I(t) of $I_j(t-1)$ we mean

$$I(t) = \{I_j(t-1, r(t)\}, \quad r(t)\varepsilon S.$$

Clearly, in all s extensions of $I_j(t-1)$ are possible. Let $I_\ell(t)$ denote an "extended" state sequence where $\ell = 1, 2, \ldots, (L_{t-1}xs)$. Now compute the a posteriori probability $P(I_\ell(t)|Z_t)$ and choose L_t (where $L_t = M$ if $L_{t-1}xs > M$, and $L_t = L_{t-1}xs$ otherwise) most likely sequences out of the

L_{t-1}xs extended sequences using the maximum a posteriori probability criterion which leads to minimum probability of error [12]. In other words, arrange $P(I_\ell(t)|Z_t)$ in a decreasing order of magnitude and select the first L_t sequences. (It should be noted that since $P(I_\ell(t)|Z_t) = [f(Z_t|I_\ell(t)) P(I_\ell(t))/f(Z_t)]$ where the denominator is common for all values of ℓ, only the numerator need be considered.) The approximate smoothed state estimate is then based on the selected structural state sequences. Therefore, (6) is modified as

$$\hat{x}(t-N|t) = \sum_{j=1}^{L_t} \hat{x}_j(t-N|t)P(I_j(t)|Z_t) \tag{17}$$

where

$$P(I_j(t)|Z_t) = \frac{f(Z_t|I_j(t))P(I_j(t))}{\sum_{\ell=1}^{L_t} f(Z_t|I_\ell(t))P(I_\ell(t))} \tag{18}$$

and subscripts j and ℓ now index the selected L_t sequences.

Remark 1. Further refinements of the algorithm to reduce the computational requirements may be carried out following [6]. The number M can be viewed as a design parameter which is selected based on the computational and storage constraints.

Structure Detection Equation (16) is modified in that the sequences on the right side of (16) are restricted to belong to the set of L_{t-1}xs extended sequences discussed earlier in this section.

Remark 2. The computational loads of the proposed approximations to the structure detection and the state smoothing problems are proportional to that of M x s elemental filters and M elemental smoothers, respectively.

V. SIMULATION EXAMPLES

In this section several simulation examples are presented to illustrate the improvement in the estimation and the detection performances due to a (fixed) lag in the estimator and the detector. For reasons of implementability, the suboptimal algorithm (proposed in Section IV) with only a "small" number of smoothers in parallel is considered.

Example 1. Consider a scalar dynamical system described by the following equations

$$x(t+1) = 1.04\, x(t) + w(t)$$

$$z(t) = C(r(t))x(t) + D(r(t))v(t), \qquad t = 0, 1, 2, \ldots$$

$$s = 2, \quad \text{i.e., } r(t) \in \{1, 2\}$$

The initial conditions are: $x(0) \sim N(30,400)$, $P(r(0)=1) = P(r(0)=2) = 0.5$. In the actual (true) system we use $x(0)=1$ for all simulation runs to generate $z(t)$. Furthermore, $\{w(t)\}$ and $\{v(t)\}$ are mutually independent zero-mean white Gaussian noise sequences with covariances $Q=0.1$ and $R=1.0$, respectively. The process $r(t)$ is modelled by a Markov chain with transition probabilities $p_{11} = 0.85$, $p_{12} = 0.15$, $p_{22} = 0.3$, $p_{21} = 0.7$. Finally, we take $D(1) = 40$, $D(2) = 1$, $C(1) = C(2) = 1$.

This example has been considered in [2], [6], [7]. The system is unstable and the state uncertainty increases with time t unless the measurements are processed. The choice of the transition probabilities implies that accurate measurement data occur only rarely.

The suboptimal algorithm was simulated for various design parameters (lag N and number of smoothers in parallel M) and operated on the same set of data. The structure detection and the state estimation performances were evaluated by averaging over 50 Monte Carlo runs. The performance measures are shown in Table 1 after averaging over 30 time stages also. It is seen that, on the average, an increase in lag N results in improved performance. However, this is not true for every time instant. The reason for this is that we are using a suboptimal algorithm and if an error is committed in selecting the M most likely structural state sequences, the error is compounded in that it first causes an error in filtering which in turn contributes to smoothing errors since filtering is a prerequisite to smoothing. It is also seen that the first few lags (in this case $N = 1$) lead to substantial performance improvement, whereas, for larger lags, any further (incremental) improvement is only marginal; this is a well-known fact [12]. It is also seen from Table 1 that increasing the number M of smoothers/filters in parallel leads to improved performance. However, the detection performance does not appear to be improved as much by increasing M with fixed $N = 0$.

Example 2. Again consider the system of Example 1 except that now $C(1) = 0$ and $C(2) = 1$; all other parameters remain unchanged. The detection and the estimation performances were evaluated by averaging over 50 Monte Carlo runs. Table 2 summarizes this information after averaging the Monte Carlo averages over 30 time stages. It is seen that increase in lag N leads to improvement in both estimation and detection performances. The choice of the transition probabilities (see Example 1) implies that most of the time the observations contain noise alone. Therefore, compared to Example 1, the estimation performance is much worse as would be expected. The performance for $M < 16$ was still worse and has not been presented.

Example 3. Now we model $r(t)$ as a semi-Markov chain. The scalar system considered is described by

$$x(t+1) = 1.04\, x(t) + w(t)$$

$$z(t) = x(t) + D(r(t))v(t), \qquad t = 0, 1, 2, \ldots$$

$$s = 3, \qquad \text{i.e., } r(t) \in \{1, 2, 3\}.$$

We have $D(1) = 100$, $D(2) = 10$, and $D(3) = 1$. The initial conditions are: $x(0) \sim N(30, 400)$, $P(r(0) = i) = 1/3$ for $i = 1,2,3$. In all simulation runs to generate $z(t)$ we use $x(0)=1$ in the true system. Furthermore, $\{w(t)\}$ and $\{v(t)\}$ are mutually independent zero-mean white Gaussian noise sequences with covariances $Q = 0.1$ and $R = 1.0$, respectively. The process $r(t)$ is modelled by a semi-Markov chain with transition probabilities of the imbedded Markov chain given by $p_{11} = p_{22} = p_{33} = 0$, $p_{12} = 0.7$, $p_{13} = 0.3$, $p_{21} = 0.6$, $p_{23} = 0.4$, $p_{31} = 0.3$, and $p_{32} = 0.7$. The probability mass functions $p_{\ell}(n)$ defined in Section IIIA.2 are assumed to be $p_1(n) = a_1 \exp[-|n-3|]$, $p_2(n) = a_2 \exp[-|n-6|]$, and $p_3(n) = a_3 \exp[-|n-8|]$ for $n \geq 0$ with a_i such that $\sum_{n=0}^{\infty} p_i(n) = 1$, $(i = 1,2,3)$.

The results of averaging over 50 Monte Carlo runs and 30 time stages are displayed in Table 3. The discussion of Example 1 pertaining to the effects of increasing N and M on the estimation and the detection performances applies here too.

VI. CONCLUDING REMARKS

The problem of state estimation and system structure detection with a fixed lag for discrete systems with abruptly changing structure was considered. The changes were modelled by a Markov or a semi-Markov chain with known transition statistics. Since the optimal solution was impractical, a suboptimal approach was proposed and its efficacy was demonstrated through several simulation examples. It was shown that (as expected) a delay in processing the observations could lead to improvements in both estimation and detection performances.

We considered only the suboptimal approach based on the detection-estimation algorithm of [6]. Other approximations are also possible following [7]; it was felt that the proposed algorithm is the most efficient computationally, given limited resources. However, for small lags, the generalized pseudo-Bayes algorithm [7] may offer a good alternative (see the results pertaining to detection and estimation with zero lag in [7]). Finally, it should also be noted that recursive optimal linear filters and smoothers (when they exist) may perform better than suboptimal non-linear filter/smoothers. Note, however, that for the system model (1)-(2), the recursive optimal linear state estimator may not exist [24]. Moreover, use of linear state estimators does not lead to the solution of the detection problem.

REFERENCES

[1] G.A. Ackerson and K.S. Fu, "On state estimation in switching environments," IEEE Trans. Automatic Control, vol. AC-15, pp. 10-17, Feb. 1970.

[2] H. Akashi and H. Kumamoto, "Random sampling approach to state estimation in switching environments," Automatica, vol. 13, pp. 429-434, July 1977.

[3] R.L. Moose, "An adaptive state estimator solution to the maneuvering target problem," IEEE Trans. Automatic Control, vol. AC-20, pp. 359-362, June 1975.

[4] C.B. Chang and M. Athans, "State estimation for discrete systems with switching parameters," IEEE Trans. Aerospace and Electronic System, vol. AES-14, pp. 418-425, May 1978.

[5] J.K. Tugnait, "Comments on: State estimation for discrete sysetms with switching parameters," IEEE Trans. Aerospace and Electronic Systems, vol. AES-15, p. 464, May 1979.

[6] J.K. Tugnait and A.H. Haddad, "A detection-estimation scheme for state estimation in switching environment," *Automatica*, vol. 15, pp. 477-481, July 1979.

[7] J.K. Tugnait, "Detection and estimation for abruptly changing systems," in *Proc. 20th IEEE Conf. Decision and Control*, San Diego, pp. 1357-1362, Dec. 1981. (Also, the final version to appear in *Automatica*, Sept. 1982).

[8] B.D.O. Anderson and J.B. Moore, *Optimal Filtering*. Englewood Cliffs, N.J.: Prentice Hall, 1979.

[9] R.A. Howard, "System analysis of semi-Markov processes," *IEEE Trans. Military Electronics*, vol. MIL-8, pp. 114-124, April 1964.

[10] R.A. Howard, *Dynamic Probabilistic Systems*, *Vol. 2*, *Semi-Markov and Decision Processes*. New York: Wiley, 1971.

[11] S.M. Ross, *Applied Probability Models with Optimization Applications*. San Francisco, Calif.: Holden-Day, 1970.

[12] H.L. Van Trees, *Detection, Estimation, and Modulation Theory, Part 1*. New York: Wiley, 1968.

[13] D.D. Sworder, "Control of systems subject to sudden change in character," *Proc. IEEE*, vol. 64, pp. 1219-1225, Aug. 1976.

[14] J.D. Birdwell, D.A. Castanon, and M. Athans, "On reliable control system designs with and without feedback reconfigurations," in *Proc. 17th IEEE Conf. Decision and Control*, San Diego, pp. 709-715, Jan. 1979.

[15] A.S. Willsky, "A survey of design methods for failure detection in dynamic systems," *Automatica*, vol. 12, pp. 601-611, Nov. 1976.

[16] H.F. Van Landingham and R.L. Moose, "Digital control of high performance aircraft using adaptive estimation techniques." *IEEE Trans. Aerospace and Electronic Systems*, vol. AES-13, pp. 112-120, March 1977.

[17] R.L. Moose, H.F. Van Landingham, and P.E. Zwicke, "Digital set point control of nonlinear stochastic systems," *IEEE Trans. Industrial Electronics* and *Control Instrumentation*, vol. IECI-25, pp. 39-45, Feb. 1978.

[18] H.F. Van Landingham, R.L. Moose, and W.H. Lucas, "Modelling and control of nonlinear plants," in *Proc. 17th IEEE Conf. Decision and Control*, San Diego, pp. 337-341, Jan. 1979.

[19] D.G. Lainiotis, "Partitioning: A unifying framework for adaptive systems, I: Estimation," *Proc. IEEE*, vol. 64, pp. 1126-1142, Aug. 1976.

[20] D.G. Lainiotis, "Partitioning: A unifying framework for adaptive systems, II: Control," *Proc. IEEE*, vol. 64, pp. 1182-]]98, Aug. 1976.

[21] D.E. Gustafson et. al., "ECG/VCG rhythm diagnosis using statistical signal analysis, 1. Identification of persistent rhythms," *IEEE Trans. Biomedical Engrg.*, vol. BME-25, pp. 334-353, July 1978.

[22] A.S. Willsky et. al., "Dynamic model-based techniques for detection of incidents on freeways," *IEEE Trans. Automatic Control*, vol. AC-25, pp. 347-360, June 1980.

[23] A.C. Sanderson, J. Segen, and E. Richey, "Hierarchical modelling of EEG signals," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. PAMI-2, pp. 405-415, Sept. 1980.

[24] M.T. Hadidi and S.C. Schwartz, "Linear recursive state estimators under uncertain observations," *IEEE Trans. Automatic Control*, vol. AC-24, pp. 944-948, Dec. 1979.

[25] N.H. Gholsun and R.L. Moose, "Maneuvering target tracking using adaptive state estimation," IEEE Trans. Aerospace and Electronic System, vol. AES-13, pp. 310-317, May 1977.

TABLE 1

No. of Smoothers M	Lag N	Average r.m.s. error in state estimation	Average probability of error in structure detection
2	0	16.02	0.15
2	1	11.68	0.12
2	2	11.10	0.11
2	5	9.84	0.10
4	0	12.75	0.13
8	0	8.07	0.13

TABLE 2

No. of Smoothers M	Lag N	Average r.m.s. error in state estimation	Average probability of error in structure detection
16	0	25.23	0.15
16	1	24.52	0.14
16	2	23.73	0.13
16	5	21.66	0.12

TABLE 3

No. of Smoothers M	Lag N	Average r.m.s. error in state estimation	Average probability of error in structure detection
4	0	4.93	0.16
4	1	4.07	0.09
4	2	3.44	0.07
4	5	2.36	0.05
2	2	12.80	0.14
8	2	3.09	0.06

STATISTICAL EFFICIENCY OF THE SAMPLE AUTOCORRELATION FUNCTION IN ARMA PARAMETER ESTIMATION

S. P. BRUZZONE* AND M. KAVEH**

*ARGO Systems, Inc., 884 Hermosa Ct., Sunnyvale, CA 94086
**Dept. of Electrical Engineering, University of Minnesota
Minneapolis, MN 55455

SUMMARY

There is presently strong interest in designing autoregressive-moving average (ARMA) spectral estimators that estimate the ARMA parameters or some function of them from several consecutive lags $c(k_1), c(k_1+1), .., c(k_2)$ of the sample ACF $c(i)$, given by $c(i) = K(N,i) \sum_{t=1}^{N-i} x_t x_{t+i}$ where $x_t, t=1,...,N$ is the observed real ARMA time series whose power spectral density (PSD) is sought, and $K(N,i)$ is a known weighting function from which the biased, the unbiased, or any other tapered version of ACF estimate is procured. We designate these estimators members of the ACF class. Attention has turned to the ACF class in response to the difficulties encountered in attempting to solve the highly nonlinear sets of equations required to get a least squares fit of the ARMA parameters to the raw data x_t. Such least squares estimates are approximately maximum likelihood for Gaussian data, however, so they are asymptotically efficient [1], i.e., they set the bound on statistical accuracy of parameter estimates from long data records. While members of the ACF class are computationally more efficient, there is a loss of information in the associated data reduction which may be large for typical cases wherein $k_1 < k_2 << N$, for it is known [2] that a finite set of lags of the sample ACF is not a sufficient statistic for estimating the parameters of an ARMA process. This results in increased statistical variation in the parameter estimates and hence greater variance in the estimated PSD. We provide a scalar measure, the relative information index [3], of the information retained in $c(i)$ relative to that originally in the raw data.

Poles and zeros of the ARMA system are assumed to occur in complex pairs. The relative information index (RII) R is defined as $R = |I_c(\theta)|/|I_x(\theta)|$ where $I_c(\theta)$ and $I_x(\theta)$ are Fisher's information matrix for the sample ACF $c(k_1),...,c(k_2)$ and the data, respectively. These are evaluated in [3], where it is necessary to take $N \to \infty$ to get tractable expressions. The parameter vector θ contains pole and zero pair magnitudes and angles, but it is shown in [3] that R is invariant under differentiable transformations of θ, so that results are valid for estimating poles and zeros, ARMA parameters, and many other reparameterizations. R is derived in [4] as a scalar assessment of the information in a statistic relative to that in the data for any multiple parameter estimation problem. It is proven there that $0 \leq R \leq 1$, with $R=1$ iff the data reduction is a sufficient statistic for θ. It is obvious from its definition that R is the ratio of the Cramer-Rao lower bound on the generalized variance [1] for estimators based on the data to that for estimators based on $c(k_1),...,c(k_2)$.

There are applications where estimates on only a partial set θ_1 of the parameters θ are desired. For example, it is often the case that only the poles of the ARMA process formed by an AR process in white noise are of interest. The partial RII (PRII) R_1, having the same properties as R, is

---

Supported in part by Air Force Office of Scientific Research Grant #AFOSR-78-3628, and National Science Foundation Grant #ECS-8105962.

derived in [4] as $R = |[I_x^{-1}(\theta)]^E| / [I_c^{-1}(\theta)]^E|$, where $[I^{-1}]^E$ denotes a matrix obtained by removing the row-column pairs involving the nuisance parameters (those not of interest) from I^{-1}.

Using the expressions derived for $I_x(\theta)$ and $I_c(\theta)$, the RII and PRII for several classes of ARMA processes were computed and compared numerically and graphically [4]. In the following we briefly mention the results obtained for an AR(2) process in white noise as a representative example of interest.

The sum of an AR(L) process and white noise is an ARMA(L,L) process, with zeros determined by pole positions and signal-to-noise ratio SNR = $E(x^2)/\sigma_n^2$. We have observed a marked drop in RII (PRII) for this process when early lags of the sample ACF are not used in the ARMA parameter estimate. Consequently, we assume early lags are retained and study the effect of truncating later ones for AR(2) estimation in white noise. Plots reveal a nondecreasing behavior of the RII and the poles-only PRII for $c(o),\ldots,c(k)$, $k \geq 2$, as SNR or k is increased or pole pair magnitude is decreased, irrespective in each case of the fixed values of the other variables. The effect of the noise is to introduce a zero pair starting at the origin for SNR = + ∞ dB and travelling outward toward the poles, cancelling them in the limit as SNR → − ∞ dB. It is this zero pair that is responsible for the insufficiency of $c(o),\ldots,c(k)$, $2 \leq k < N-1$, for estimating parameters of an AR(2) process in noise.

This appears to be a general result that to a large extent the zero placement determines R. The best achievable performance of estimates based on the sample ACF can therefore be predicted rather well from a knowledge of the pole-zero constellation for the ARMA process. Simply stated, deterioration accompanies proximity of zeros to the unit circle. Thus, narrowband AR signals in noise, PSD having closely interfering AR peaks, and dominant zeros (i.e., PSD with antiresonances) all require a large value of k_2 to even allow the possibility that the performance of a spectral estimator based on the sample ACF rival that of data-based least squares ARMA estimators.

REFERENCES

(1) H. Cramér, Mathematical Methods of Statistics, Princeton University Press, 1974.

(2) M. Arato, "On the sufficient statistics for stationary Gaussian random processes," Theo. Prob. Appl., Vol. 6, No. 2, pp. 199-201, 1961.

(3) S. P. Bruzzone, "Information considerations in the design of autocorrelation-based ARMA parameter estimators," Ph.D. Dissertation, University of Minnesota, Minneapolis, 1982.

(4) S. P. Bruzzone and M. Kaveh, "A criterion for selecting information-preserving data reductions for use in the design of multiple parameter estimators," accepted for publication in the IEEE Transactions on Information Theory.

(5) G. E. P. Box and G. M. Jenkins, Time Series Analysis: Forecasting and Control, San Francisco: Holden-Day, 1970.

REAL TIME SYSTEMS: AN APPROACH TO COMPUTER AIDED DESIGN OF HARDWARE AND SOFTWARE

ALAN A. ROSS, HERSCHEL H. LOOMIS, JR., GEORGE G. POLLOCK
Computer Science Dept, Naval Postgraduate School, Monterey, Ca.
Electrical Engineering Dept, NPS, Monterey, Ca.
Lawrence Livermore National Laboratory, Livermore, Ca.
respectively

The historical approach to design of computer based systems has been to purchase or at least choose a set of hardware and then write software to drive this hardware to accomplish the system tasks. If the hardware costs are very large compared to the software costs, then this is a feasible approach. Today, however, the microprocessor era has reversed these cost relations, and a small computer may be easily purchased for the same price as a few hours of a programmer's time. The question that we are attempting to answer is: "Is it practical to design and describe the function exactly, then automatically produce the software and only sufficient hardware to support this software?" We have demonstrated that this approach is possible, and this paper will describe some of the feasibility tests we have conducted, the current state of our effort, and our future plans.

The compiler is a computer aided design tool which takes a high-level or abstract expression of a system design and automatically translates it into a program using the set of machine language codes for a particular machine. Our approach to the problem set forth above is to extend the concept of a compiler in three ways. First, instead of generating code for a single target machine, the compiler will generate code for any of a number of machines, and it will choose the target machine based on the characteristics of the problem to be solved. Second, it will also automatically generate the listing of the hardware necessary to support the software. Third, the design system will generate a monitor which will properly sequence the code so as to guarantee that the program runs correctly, that is, for example, it meets any timing constraints.

The first application of this design system is to the design of real-time control systems. M.N. Matelan[1] described the initial concept of automating real-time controller design by abstracting the controller description to a set of contingency/task pairs. A contingency is an occurrence, either external or internal to the controller, that requires the execution of a task in the controller. The controller is described by listing the set of contingencies it must recognize, the task it must perform in response to each contingency, and the time constraints on contingency recognition and task execution. Once the controller is described by the set of contingency/task pairs it must control, it can be realized by realizing each contingency as a function, each task as a procedure, and by using the timing information to construct a real time monitor. We have since discovered that the problem must also be partitioned into sections corresponding to operational modes. The example in this paper includes an initialization section and a run section.

As presently implemented, the design system accepts as input a contingency/task model of the controller to be designed, and after testing to insure that a realization is possible, generates the hardware lists and assembly language code for a microprocessor-based controller.

Figure 1 shows a block diagram of the design system. It is library driven, with a module in the library for each of the microprocessor hardware families. The library entries include a software routine to realize each of the possible primitives of the c/t pair description, the hardware necessary to support each of these software routines, and timing data which indicates how long each of these realizations takes to execute. The design system accumulates this timing data as it searches the library for the primitive realizations. If a particular primitive cannot be implemented with a particular hardware family, the primitive realization will not be present in the library, and the design system will report this failure and shift to another hardware family to attempt a realization. If all of the primitives of a problem are successfully realized, the timing analyzer performs a timing analysis to determine if all of the timing constraints of the problem are met by the realization, and to construct a monitor which will control and sequence the implementation.

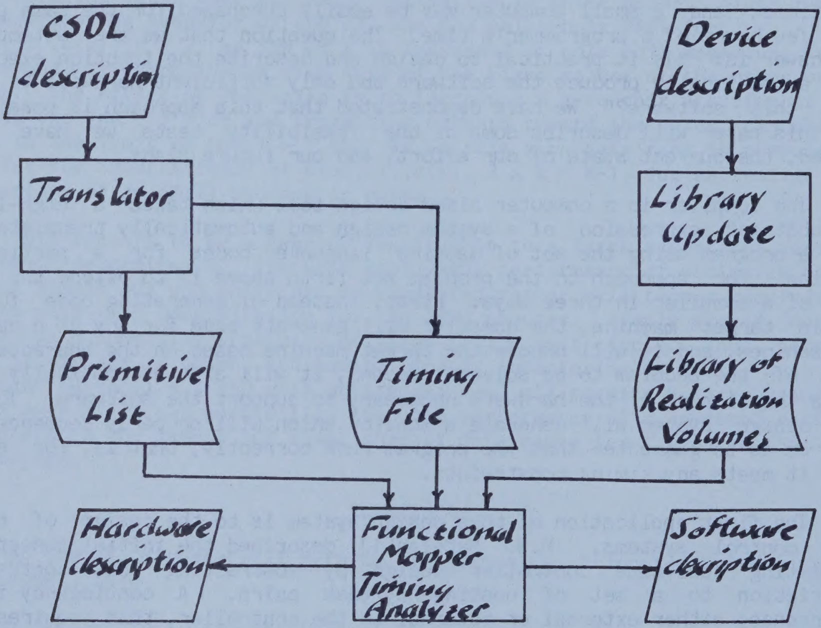

CSDL description
Device description
Translator
Library Update
Primitive List
Timing File
Library of Realization Volumes
Hardware description
Functional Mapper Timing Analyzer
Software description

Figure 1. CSDL System

We have examined a number of realistic control problems with the improvement of the design system and CSDL in mind. One such system is the controller for the fuel injection system for a DATSUN 280Z automobile. The control system achieves high engine performance while satisfying American pollution and fuel economy requirements by precise control of fuel flow as a function of a number of engine parameters.

Fuel is injected into the intake manifold in the vicinity of each intake valve by an electrically operated injector valve. Each of these valves injects a fuel charge each time the distributer breaker points open. The width of the pulse supplied to the valve determines the amount of fuel injected.

The fuel pulse width is calculated as follows:

```
    W = (K1 * F) * WTMP * ATMP * TH * ALT
where:
    W is the width of the fuel injection pulse,
    K1 is a proportionality constant
    F is the air flow measurement;  air flow
        is controlled by the accelerator,
    WTMP is the water temperature correction factor,
    ATMP is the air temperature correction factor,
    TH is the throttle limit correction factor, and
    ALT is the altitude (pressure) correction factor.
```

The correction factors are calculated as follows:

```
    WTMP = 1 + ((70 - TW)**2)/10000.
    ATMP = 1 + (20 - TA)/400.
    ALT = 0.94 if pressure exceeds 660 mm Hg,
          1.00 if pressure is less than 660 mm Hg.
    TH  = 0.00 if throttle closed,
          1.27 if throttle full open, or
          1.00 otherwise.
where:
    TW is the water temperature, and
    TA is the air temperature in degrees C.
```

This represents a simplification of the complete control system which includes special behavior during engine start.

The CSDL description of the controller is organized into two sections, an initialization section and a run section. The initialization section executes most of the tasks once to establish initial values for all of the control variables. The run section is organized into six contingency-task pairs, one each for calculation of W, KA, TH, ATMP, WTMP, and ALT. The last 3 contingencies involve very slowly changing variables, and hence their period can be comparatively long, 240 seconds. The throttle can move in fractions of a second, so the TH contingency has a period of 200 milliseconds. The KA contingency calculates the product of the outputs of all the previously mentioned tasks, hence it is assigned a period of 100 ms. Finally, the output pulse width must be updated at least every 35 ms.

The CSDL description of the controller is shown in Figure 2. The nature of this description should be readily apparent from the description itself and the previous discussion.

Once the CSDL description has been compiled, a set of software and hardware descriptive primitives is produced. A portion of the primitive description for our controller is shown in figure 3, illustrating the way the functions CW and CALT and the tasks TW, and TALT are compiled.

The complete primitive file is then processed by the functional mapper and the timing analyzer, in this simple case producing a feasible realization using the 8080 family. Figure 4 presents portions of the software that realizes our controller, in particular the function CW and a portion of the task TW.

```
IDENTIFICATION:

    Designer: Pollock/Loomis
    Date:  23-Jun-82
    Project:  280Z Fuel Injection Controller

ENVIRONMENT:

    Input:IF,8,RESTRAN;THR,1,TTL;THF,1,TTL;AT3,8,TEMP;
        ATR,1,TTL;WTR,1,TTL;WT3,8,TEMP;S,1,SWITCH;
        STSW,1,TTL;
    Output:W,8,TTL;
    Arithmetic:KA,24:REAL; K1,24:REAL; ALT,24:REAL; WT,24:REAL;
        AT,24:REAL; TH,24:REAL; OLDS,1:BINARY;

CONTINGENCY LIST:

  SECTION: INITIAL;

    "Initialization Contingency"
    when START:100MS do SETUP;

  end "INITIAL";

  SECTION: RUN;

    "Control Fuel pulse width contingency"
    when CW:35MS do TW;

    "Altitude, Water Temp and Air temp factors contingency"
    when CKA:100MS do TKA;

    "Throttle sensing contingency"
    when CTH:200MS do TTH;

    "Air temp factor contingency"
    "AT=(1+(20-T)/400)"
    when CATMP:240 do TATMP;

    "Water temp factor contingency"
    "WT=WT1+(70-T)**2/1E4"
    when CWTMP:240 do TWTMP;

    "Altitude correction factor contingency"
    "If p < 26 inHg then ALT:=1 else ALT:=.94"
    when CALT:240 do TALT;

  end "RUN";
```

Figure 2. CSDL Description of Fuel Injection Controller

```
PROCEDURES;
    function START;
        binary START,1;
        sense STSW;
        if STSW=1 then START:=1 else START:=0;
    exit START;

    function CW;
        binary CW,1;
        CW:=1;
    exit CW;
        .
        .
        .
    function CALT;
        binary CALT,1;
        sense S;
        if S/=OLDS then CALT:=1 else CALT:=0;
    exit CALT;

    task SETUP;
        K1:=10.5;
        do TTH;
        do TATMP;
        do TWTMP;
        do TALT;
        do TKA;
        switch section to RUN;
    exit SETUP;

    task TW;
        sense (IF);       "Resistive transducer for air flow"
        W:=IF*KA*K1;
        issue W;
    exit TW;

    task TKA;
        KA:=ALT*WT*AT*TH;
    exit TKA;
        .
        .
        .
    task TALT;
        sense (S)      "altitude switch"
        OLDS:=S;
        if S=1
           then ALT:=.94
           else ALT:=1.
        fi;
     exit TALT;
```

Figure 2. CSDL Description of Fuel Injection Controller

One of the most difficult problems with the development of the design system has been the automatic generation of the monitor. We have examined three possible concepts, each to a different depth. First, we have implemented a polled monitor strategy, where the controller's basic function is to test every contingency in a sequence which is determined by the design system based on the timing limits specified for each c/t pair. We have also done some theoretical development of an interrupt driven monitor, where each contingency is implemented in hardware so that it generates an interrupt every time the c/t pair requires service. Third, we have started to consider a time division monitor which will be driven by a real time clock in

```
T. Generated For: CW
S.EVERY          (CW::)
S.VAR            (CW:8)
T. Generated For: CALT
S.PROC           (CALT::)
S.SENSCONTAC     (S,SBUF:)
S.NE             (CALT,S,OLDS:8,8,8)
S.EXITPROC       (CALT,0::)
S.VAR            (CALT:1)
T. Generated For: TW
S.PROC           (TW::)
S.SENSECOND      (IF,1000:1000)
S.FLOAT          (F,IF:8)
S.FMUL           (TW1,F,KA:24,24,24)
S.FMUL           (W,TW1,K1:24,24,24)
S.FIX            (IW,W:8)
S.ISSUECOND      (IW:8,128)
S.EXITPROC       (TW,CW::)
S.VAR            (IF:8)
S.VAR            (F:24)
S.VAR            (TW1:24)
S.VAR            (W:24)
S.VAR            (IW:8)
T. Generated For: TALT
S.PROC           (TALT::)
S.SENSCONTAC     (S,SBUF:)
S.ASSIGN         (OLDS,S:8,8)
S.EQ             (T1,S,C001:8,8,8)
S.JMPF           (T1,L001:8)
S.ASSIGNCONS     (ALT,0.94:24,24)
S.JMP            (L002)
S.LOC            (L001)
S.ASSIGNCONS     (ALT,1.0:24,24)
S.LOC            (L002)
S.EXITPROC       (TALT,CALT::)
S.CONS           (C001,1:8)
S.VAR            (T1:8)
```

Figure 3. Primitive List for CW, CALT, TW, TALT

```
;******  Contingency for CW
        ; Dummy procedure for every-period type contingency
        @CW:    nop             ;dummy function entry point
                mvi     a,1     ; force function value
                sta     CW      ;  to true value (1)
                ret             ; return to monitor
        cw:     db      0       ;8 bit variable CW
;****** Procedure TW
        @tw:    nop             ;entry point for TW
                in      0       ;sense environmental data
                sta     IF      ; and store in IF
        ;convert integer IF to floating point F
                lda     IF      ;load argument lsbyte
                out     1
                cpi     0       ;set flags
                jp      $+8
                mvi     A,377B  ;negative, extend sign
                jmp     $+2
                xra     A       ;positive, clear the msbyte
                out     1       ;load the argument dummy msbyte
                mvi     a,35B   ;output command
                out     2
                in      1
                sta     F+3     ;unload result
                in      1
                sta     F+2
                in      1
                sta     F+1
                in      1
                sta     F
                in      2       ;check error code
                ani     36B     ;mask the error code field
                jz      $+8     ;no error, continue
                push    PSW     ;  error, call error routine
                mvi     A,11B   ;  identify source of error
                call    error
        ;multiply f.p. F by KA to get TW1
                .
                .
                .
```

Figure 4. Realization of CW and the first primitives of TW

the controller, and which will allocate the processor to the highest priority task for each time slice.

As implied earlier, the library of primitive realizations is one of the key parts of the design system. This library is also one of the most difficult portions to implement. Figure 5 shows an example of a software primitive and a hardware primitive from an existing library volume.

```
S.NE            (RSLT,ARG1,ARG2 :0,8,0,8,0,8 :20,80,22,,,,)
COM PRIMITIVE TO SET RESULT TRUE IF ARG1 NOT EQUAL ARG2, ELSE FALSE
COM LIST=RESULT, ARGUMENT1,ARG2: PRECS : STOR, TIME, EXT,CALC,INCL,
BEGIN STEXT
        LXI     H, <RSLT>        ; PRESET <RSLT>
        MVI     M, 1             ;   TO TRUE (1)
        LDA     <ARG1>           ; FETCH FIRST ARGUMENT
        LXI     H, <ARG2>        ; POINT TO SECOND ARGUMENT
        CMP     M                ; MAKE THE COMPARISON
        JNZ     $+6              ; SKIP IF NOT EQUAL
        LXI     H, <RSLT>        ;  ELSE POINT AT <RSLT>
        MVI     M ,0             ;  AND CLEAR IT.
ENDTEXT
CALC ROMPTR = ROMPTR + 20


H.ISSUECOND (SIGNAM:0,8,0,128: 2,500,1,,,,)
COM PRIMITIVE TO DEFINE CONDITION-TYPE OUTPUT HARDWARE
COM LIST = SINK,:MIN/MAX-OUTPUT-LINES, MIN/MAX CONDS:
COM      LATENCY, POWER, CALC, INCL, ADDR
BEGIN HTEXT
  CONDITION-MODE OUTPUT INTERFACE HARDWARE TO ISSUE SIGNAL: <SIGNAM>
    DEVICE:  INTEL 8212 8-BIT I/O PORT, IC <ICN>
    CONNECTIONS:
     PINS 3,5,7,9,16,18,20,22 (DI(1:8)) = DB(1:8)
     PINS 4,6,8,10,15,17,19,21 (DO(1:8)) = <SIGNAM>(1:8) ; IF 8 REQUIRED
     PIN  2 (MD) = +5V
     PIN 11 (STB) = GND
     PIN  1 (DS1-BAR) = WR-BAR
     PIN 13 (DS2) = OUT .AND. (DECODE A(0:7) VALUE <OUTPRT>)
     PIN 24 (VCC) = +5V
     PIN 12 (GND) = GND
ENDTEXT
CALC ICN = ICN+1
```

Figure 5. Sample Primitive Realizations

Each of the primitive realizations in the library must contain two classes of information. First is information about the primitive for the design system. The most critical information is the duration of the primitive execution, but it is also important to know the number of memory locations that the primitive uses, whether that memory is read/write (for variables) or read-only (for constants and code), and (for hardware primitives) the number of chips that constitute the realization. The second type of information is the code or chip list and interconnections - the data that is generated as the system designs a controller. We will examine the sample primitives of Figure 5 to determine how the library entries provide the needed information.

S.NE is a software primitive to perform a test on two arguments and set a result flag to true if the arguments are not equal, and to false if the arguments are equal. The first line includes the title of the primitive (S.NE), the dummy argument names (RSLT, ARG1, ARG2), the limits on the precisions of these arguments , and three resource counts. The resource counts are the number of bytes of memory required by the primitive (20), the number of timing periods required to execute the primitive (80), and the number of memory references made by the code in the primitive (22). The system uses the precisions of the arguments to choose a realization that matches the CSDL request. The resource counts are used to determine the speed and size of the realization. The memory reference count is included because some implementations use memories which are slower then the main processor and processor wait states must be inserted to match the speeds. These wait states must be considered when the timing analyzer calculates the worst case timing.

The next two lines of S.NE are comment lines, containing an abbreviated version of the preceding paragraph. The next block of lines, delimited by the BEGIN STEXT and ENDTEXT lines, is the source code for an 8080 assembler, and is the information contained in the listing generated by the system when this primitive is invoked. Note that the dummy arguments appear enclosed in "<" and ">". The system automatically replaces every occurrence of the dummy arguments with the corresponding actual argument. Finally, the last line of the primitive realization is a CALC line. It is a message from the realization to the system to modify a global variable. In this case, the global is a pointer to the ROM which contains the program code.

The structure of the hardware primitive is very similar to the software primitive. The title line contains the title, a dummy argument, limits on the number of lines for this chip and the number of addresses which can be decoded for it, and the latency (delay factor) and power consumption of the chip. These values are accumulated by the design system as the realization is generated, and tested to see if this particular realization can be used. In this case, the actual text describes the 8212 integrated circuit and its interconnection into a system.

The system does not automate the design of a controller, it automates the process of selecting a subset of the primitive realizations to meet the problem requirements. The creative portions of the design process are moved to the generation of the library primitives and to the generation of the problem description in the higher order language. That's the good news; the bad news is that all possible primitives must have at least one realization in each working library, or else some CSDL designs will not be realizable. Thus a tremendous amount of high quality design work is necessary to describe each primitive. This is compounded by the desirability of having multiple realizations of some primitives, to give realization alternatives within volumes. Real arithmetic is one place where such multiple realizations might be desirable; the simplest might involve software routines, the more complex special floating point chips to be incorporated in the realization.

A hardware descriptive language (HDL) appears attractive as a way for describing new hardware capabilities and timing parameters. Primitives would also be described by the HDL, and an automatic library builder would simply take the capabilities realized by the chip and attempt to use it to realize whatever primitives it can.

This is a difficult problem, and so far no tangible results have been achieved. There is substantial payoff for success, however, not only in the CSDL library problem, but also in the more general problem of how to describe new hardware components to designers and CAD programs such as design translators or simulators. Also, most HDL's do not provide the capability to describe the functions of a complex LSI integrated circuit such as a microprocessor, nor to describe the timing parameters of even simple devices.

Our current research and development efforts on the system include:

1) Installation of the current system on a VAX computer.

2) Incorporation of a commercial data base system to simplify the operation and to improve portability.

3) Selection of a standardized problem description language, either CSDL or some other choice.

4) Implementation of a compiler to translate the descriptive language into primitives.

5) The inclusion of an interrupt driven monitor strategy in the software alternatives.

6) The use of an HDL to describe functions of available devices for the library.

We believe that our research has demonstrated the feasibility of an automated design system for dedicated microprocessor based systems. The realization of the first 5 of the above research objectives will result in a valuable design tool for the hardware and software of such systems. The realization of the 6th, more difficult, goal will not only greatly enhance the capability of this system, but also contribute to many other efforts in CAD and simulation of electronic designs.

---

[1] M.N. Matelan, *Automating the Design of Dedicated Real Time Control Systems*, Preprint UCRL-78651, Lawrence Livermore Lab, 21 August 1976.

A GRAPHICAL INTERCONNECTION LANGUAGE AND ITS APPLICATION TO CONCURRENT AND REAL-TIME PROGRAMMING

BRUCE W. WEIDE, MARK E. BROWN[1], JOSE A.S. ALEGRIA, and GLENN MEYER
Department of Computer and Information Science
The Ohio State University
Columbus, Ohio 43210

Abstract - STILE (STructure Interconnection Language and Environment) is a comprehensive project that involves construction of a programming environment based on a graphical interconnection language. This report presents an introduction to the STILE language by giving an overview of the language components, informal operational semantics for these components, and examples of their use. One of the main application areas of STILE is in real-time and embedded systems, which often include special-purpose hardware components. The language was designed to improve software development technology, but also was influenced substantially by architectural issues.

GOALS AND MOTIVATION

We have undertaken a project to examine novel solutions to two serious problems of software development:

- The need for interconnection languages to encourage and facilitate development of reusable software components, and to permit construction of larger programs from those tools by non-expert programmers.
- The need for language support for real-time applications in a form that can be used readily by engineers without special computer science training.

The first problem is best described by resorting to an analogy to the situation of a decade or two ago. Then, most applications programming was moving from assembly language to high-level language programming. It had become apparent that software must be human-readable in order to improve reliability and to facilitate maintenance. Great emphasis was placed on developing language features to suppport "structured programming", and there can be little doubt regarding the success of these efforts.

It is now equally obvious that program parts must be written in a way that permits them to be reused in tasks other than those for which they were originally intended. The UNIX concept of filters and pipes is one approach that seems to have evolved from this realization [Mashey 76], [Kernighan 81]. All indications are that we have both the technology and the motivation to move up to this next level, where the primitives are much more sophisticated operations (such as UNIX filters), and the language problems involve interconnection of the available primitives. One of the factors leading to this conclusion is the proliferation of personal computers and workstations, the ultimate result of which will be that there

---

[1] Also with Bell Laboratories, Columbus, Ohio.

Work supported in part by the Graduate School, The Ohio State University.

are very many professionals who are not trained in computer science who must write software to do their work. Thus, one aspect of this problem involves making the interconnection language as easy to use as possible, even for a non-expert.

There has been little work on novel approaches to language mechanisms to be used in this higher-level domain. Even Ada, which claims to support tool-building through the use of "packages" and other constructs, is still only a perturbation of tried-and-true block structured languages that preceded it. The first problem, then, is to provide language support for development of reusable software components and for their interconnection into useful applications programs, and to make the language elegant and simple enough to be used by non-computer scientists.

We see the second, very much related, problem as being at least as disturbing, since it is in the important area of real-time programming. Most of the world has made the step from assembly language to higher-level languages, but in the real-time domain there is still a tremendous amount of programming being done in assembly language! Here, in an area where reliability is perhaps more critical than anywhere, the weakest of programming practices are most widespread. There have been attempts to extend traditional high-level languages to process control and similar applications (for example, [Wirth 77] and [Schutz 79]), but none has achieved a substantial penetration of the market. According to engineers in industry, assembly language remains by far the most prevalent language in real-time applications programs, with (of all things) Fortran running second.

It is apparent that part of the problem here stems from the fact that the engineers who are responsible for designing and developing process control software are not trained in computer science concepts, including interprocess communication, synchronization, mutual exclusion, and the like. They are comfortable with Fortran and assembly language, and are ill-equipped and reticent to adopt higher-level special purpose languages whose primary differences from Fortran are constructs for dealing with problems they do not even recognize. The requirement is to provide language support at a high enough level that details of communication and synchronization are not visible to the engineer who writes such programs.

In summary, then, there are two problems. One involves creation of software from components by specifying their interconnection at a very high level. The other is, in some sense, a special and slightly more complex case, since it involves essentially the same task, but also must accommodate explicitly notions of timing, concurrency, communication, and synchronization, that are not particularly familiar to those who must construct the final application software systems. The purpose of this paper is to introduce the main aspects of the STILE language, and to see how the language design problem interacts with design of the programming environment and potential special-purpose architectures that might be needed in embedded systems.

BACKGROUND

For the past two years, we have been investigating the potential for using a graphical interconnection language with hierarchical structure as a means of attacking both these problems. At once, such a language is useful for describing how reusable software components can be connected together to do useful work, and for hiding low-level details of real-time programming from the application programmer and engineer. Our initial attempt to design such a language has led to development of STILE (for STructure Interconnection Language and Environment), which is a comprehensive system designed to facilitate general interconnection of software components. Being a graphical language and a comprehensive programming environment, STILE permits the programmer or other less expert user to "see" exactly how the parts interact, and thereby to take full advantage of human visual capabilities that are unmatched by machine.

The idea of a graphical interconnection language is not new (see, for example, [Dennis 74], [Milner 77], [MacQueen 79], [Sharp 79], [Jensen 79], [Yamano 81]). STILE is different in several respects from previous similar projects, though. It is designed to address a wide range of problems, and to include a customizable base of program components in a variety of applications areas where non-experts might be expected to want to program. Most importantly, integrated with the STILE graphical language is a friendly programming environment that can help direct the application programmer to a final solution by channeling efforts to a certain design discipline and style of development.

It should be noted that, from the initial stages of the project, STILE was intended to address problems associated with software design for real-time process control systems [Brown 80]. Despite its origination as a real-time language (still one of the main application areas of interest), STILE also appears to be useful in a more general setting, so the semantics of the graphical language components are described without regard for their possible use in real-time programming. Nevertheless, the power and convenience of STILE are vividly deomonstrated in the real-time area, which presents significant problems (involving timing and the necessity for very low-level routines) that best illustrate the importance of STILE′s uniform presentation methods.

THE STILE LANGUAGE

The STILE language (called just "STILE" here) is a language for specifying how reusable program components are to be interconnected to create a more complex program. The underlying virtual machine on which STILE graphs "run" is quite simple. A (STILE) graph is an undirected graph in which the nodes are called boxes and the edges are called links. Each box has some ports for communication with other boxes via links. The graph structure is static; that is, it is fixed at construction time[1] and cannot be changed at execution time.

In short, a box can be simple, in which case it contains sequential code to perform some computation and to test and manipulate certain characteristics of its ports; or it can be composite, or a subsystem, in which case it is a subgraph consisting of any number of boxes connected by links, that operates in a larger graph just as though the subgraph were substituted for it in "macro" fashion. (More details about boxes are given below.) Links are the binding mechanism for creating graphs from boxes.

---

[1] By "construction time", we mean what essentially amounts to a combination of program preparation time, plus compilation time, plus linkage time, in a more conventional programming system.

Links

Links attach to ports, and there may be at most one link connected to any port of any box. Each link has the following characteristics:

- a type, which is the data type of the value of the link;
- units, which give the interpretation of the value using ordinary engineering units;
- a state, which is either empty or full;
- a value, which may be written or read by the boxes it is attached to.

All these characteristics are specified at graph construction time. Type and units characteristics are static, so that type-checking and units-checking are the responsibility of the graph construction system (programming environment), not the run-time system. The state and value characteristics, on the other hand, are affected by the execution-time actions of the boxes at either end of the link. Initial values for state and value are specified at construction time, but may change during execution.

Ports

Each box has a number of ports, each of which has exactly the same characteristics as a link: type, units, state, and value. Again, type and units are static, while state and value may change during execution but are assigned initial values when the box is created.

Ports are associated with links during graph construction, with the obvious correspondences of characteristics. Some ports may not be connected to a link, in which case the initial values for the characteristics are those assigned for that port in the description of the box. For ports connected to a link, there are two cases. If the other end of the link is not connected to a port, then the port it is connected to inherits the initial state and value characteristics of the link as its initial values; of course, the type and units characteristics of the link must match those of the port. If the other end of the link is connected to another port, then both ports inherit the initial state and value of the link as their initial values, and both ports must have the same type and units as the link.

The ports of a given box are known by name within that box, but to no others. The links are known to no box, but are merely the "glue" that binds ports (hence boxes) together. Links, then, may be considered to be temporary entities known and useful only at graph construction time. They are the mechanism for creating a graph from boxes. Only ports are available to boxes for communication at execution time. This implies that a box does not know what other boxes it is connected to, and is therefore constrained to be an independent "module" by the nature of the STILE language itself. This is different from Hoare's CSP [Hoare 78], for instance, in which the communicating processes must know each others' names.

Boxes

As mentioned above, there are two types of boxes: simple boxes and composite boxes, or subsystems. A composite box is (internally) a graph in its own right that, when connected in a larger graph, works just as though the graph it comprises were connected into the larger graph as in a "macro" expansion.

We have borrowed some notation from Dijkstra's guarded commands [Dijkstra 76], with only minor modification, to informally describe the semantics of a box. Each box (of either type) in a STILE graph can be thought of as operating concurrently with all others in the graph, and executing the code shown in Figure 1.

```
do
    $FC_1$: $S_1$;
    $FC_2$: $S_2$;
        .
        .
        .
    $FC_n$: $S_n$
od.
```

Figure 1: Actions of a Box in STILE

Each box can be thought of as having its own serial processor that performs its computations. The notation in Figure 1 means that this processor continually monitors the firing conditions FC_i, which are boolean expressions relating only to the states of the ports of the box and to the current value of **time** (a built-in variable holding the current time). When one of the FC_i is **true**, the box's processor chooses (arbitrarily) to execute the action S_i corresponding to one of the **true** firing conditions. It then returns to monitor the FC_i again, proceeding as before, and never stopping.

The firing conditions FC_i may consist of boolean expressions involving **time**, and the system functions Full (port), which returns **true** if the link connected to "port" is full and **false** if it is empty, and Empty (port), which does the opposite.

The actions S_i may involve any of the ordinary assignment and control statements of a typical sequential programming language, plus invocation of the system function Read (port), which returns the value of the link connected to "port"; the system procedure Write (port, value), which sets the value of the link connected to "port" to "value"; and SetState (port, state), which sets the state of the link connected to "port" to "state".

A simple box may have essentially arbitrary firing conditions and may perform arbitrary actions, and the composition rules that allow subsystems to be inserted into larger graphs preserve the semantics of simple boxes for composite boxes. Therefore, the user need not know whether an existing box is simple or composite when constructing a graph using it, and in fact probably should not know in most cases. This means that subsystems can be reused without regard to their internal structure.

A SIMPLE EXAMPLE

As an example of a convention one might adopt regarding the meanings of ports and the uses of states and values, consider a box that multiplies two numbers. It has three ports (call them x, y, and z), and performs the function z := x * y whenever ports x and y are **full** and port z is **empty**.

Here, we have adopted the (common) convention that ports are directional, and therefore can be partitioned into <u>input</u> ports and <u>output</u> ports. (The programming environment can enforce this convention, so that the run-time system is cleaner.) The state of a port is used to provide control information; namely, **full** means that data is "available" on the link. With this convention, our multiplication box should also, of course, "consume" the values from x and y by setting the states of those ports to **empty**, and indicate that it has "produced" a new value on z by setting that port to **full**. The firing rule for this box is a popular one; it is, for instance, the firing condition in Dennis′ data flow language [Dennis 74].

```
port x: (numeric, , , );
port y: (numeric, , , );
port z: (numeric, , , );
do
    Full (x) and Full (y) and Empty (z):
            Write (z, Read (x) * Read (y));
            SetState (x, empty);
            SetState (y, empty);
            SetState (z, full)
od.
```

Figure 2: Code for "Multiply" Box

Code for this box is shown in Figure 2. We have given the port characteristics in the form (type, units, initial state, initial value). The blank field for units indicates that the operation of the box is independent of the units (and therefore that any units given for the attached link will match). The blank fields for state and value indicate that the initial state and value must be inherited from the attached link, and that there must be an attached link. (It is up to the programming environment to enforce this restriction, too.)

Another useful box is one that acts as a "fork" for data values, copying the value from its single input port to both its output ports. Code for this box is shown in Figure 3.

Now suppose we would like to create a box that squares its input, using the same firing rule as above. This (composite) box can be created by making its input port correspond to the input port of a "fork" box, by connecting

```
port in: (numeric, , , );
port out1: (numeric, , , );
port out2: (numeric, , , );
numeric temp;
do
    Full (in) and Empty (out1) and Empty (out2):
        temp := Read (in);
        Write (out1, temp);
        Write (out2, temp);
        SetState (in, empty);
        SetState (out1, full);
        SetState (out2, full)
od.
```

Figure 3: Code for "Fork" Box

the output ports of the "fork" to the input ports of the "multiply" box, and by making the output port of "square" correspond to the output port of "multiply". Note that two links are required here for internal connections between the outputs of the "fork" and the inputs of the "multiply", and that initial states and values <u>must</u> be specified for them. The states should be **empty** if the "square" box is to behave properly, and the initial values of these links are irrelevant. The resulting (composite) box is shown in Figure 4.

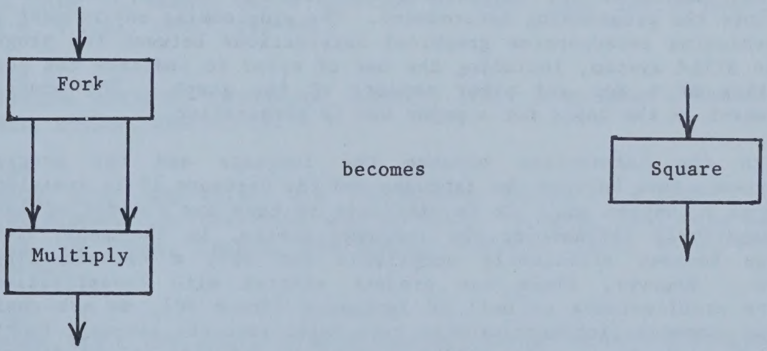


Figure 4: A Composite Box: "Square"

This example is somewhat simple, but it illustrates the potential for creating a relatively small set of simple boxes, upon which can be constructed much more elaborate composite ones. An actual useful system for real-time applications, for instance, would include primitive boxes for dealing with devices such as A/D and D/A converters, as well as computational boxes.

One additional need in a real-time application is a means for dealing with time itself. Of course, since **time** is built into STILE, this could be left up to the programmer, as it is now in most real-time languages [Wirth 77], [Wegner 79], [Allichin 80], [Anaconda 81]. On the other hand, it is easy and extremely valuable to provide a set of boxes that perform the most common time-oriented functions required in real-time programming. All can be constructed from one simple box, which we have dubbed "clock", and which we introduced in [Brown 80]. The code for a clock box requires only about 15 lines in STILE. From this simple box, many interesting composite boxes can be constructed, including watchdog timers, sampling timers, one-shots, and other important timing mechanisms of real-time systems. It is expected that a library, or database, of such boxes would be constructed once, and that the boxes in it would appear as primitives to engineers using the system for real-time programming applications. This issue is addressed by the programming environment, not by the language itself, except to say how subsystems are to work.

THE PROGRAMMING ENVIRONMENT AND ARCHITECTURAL ISSUES: AN OVERVIEW

While we believe that with at most minor modifications, STILE graphs will be powerful enough to encompass a wide class of concurrent and parallel programming problems, we are equally certain that by limiting ourselves to a particular application area, we can construct a more useful design system by incorporating domain-dependent information and enforcing a rather rigid programming discipline. Thus, at the same time that we are trying to make the STILE language as general as possible, we will also be trying to make its programming environment flexible, so that it can be modified to accommodate other potential application domains. This involves a careful balancing act in which the methodological aspects of program design for the real-time domain do not contaminate the language design, but rather are built into the programming environment. The programming environment we are now developing incorporates graphical interactions between the programmer and the STILE system, including the use of color to indicate the state of completion of a box and other aspects of the graph. The programming environment is the topic for a paper now in preparation.

As with the interaction between the language and the programming environment, that between the language and the hardware it is envisioned to run on is a complex one. It is difficult to keep one's model of hardware from negatively influencing the language design, in the sense that the language becomes efficiently compilable for only a certain class of systems. However, since our project started with investigations of hardware architectures as well as languages [Brown 80], we are confident that the communication mechanism we have built into the language (ports and links) is suitable for implementation both in centralized and distributed systems.

For technical reasons having to do mainly with timing [Lamport 78], distributed real-time systems are somewhat more difficult than they seem at first glance. But since communication is explicit and localized in STILE graphs, and does not need to be inferred from ordinary sequential code as with other systems, running in a distributed environment is not difficult. All communication links involve only two parties, so the job of coordinating the communications among several parties is not encountered [Aguilar 82].

One way to implement a distributed STILE run-time system is, as suggested above, to distribute the run-time support throughout an arbitrary distributed system. Another, to which we have already devoted considerable effort [Brown 80], [Brown 81], is development of a dedicated communication system designed specifically to support STILE graphs. This is a reasonable alternative because real-time systems are usually dedicated anyway, so the hardware can be designed along with, or even after, the software.

SUMMARY

The STILE project involves both a language component and a programming environment. The language is a graphical one, in the sense that sequential processes ("boxes") are joined together via "links", and the design process proceeds via graphical interactions between the user and the STILE programming environment. We have approached the design of these components in parallel with each other and with various hardware architectures on which the resulting programs might run, and have, we believe, been able to avoid some serious pitfalls encountered by other language efforts in the real-time domain.

REFERENCES

Aguilar 82
AGUILAR, L., AND WEIDE, B.W. Efficient broadcast acknowledgement for store and forward packet switching, in *Proc. 15th Hawaii Intl. Conf. on Syst. Sci.*, IEEE, Honolulu, HA, Jan. 1982, pp. 499-508.

Allichin 80
ALLICHIN, J. Modula and a question of time, *IEEE Trans. Softw. Eng. SE-6*, 4 (Jul. 1980), 390-391.

Anaconda 81
ANACONDA ADVANCED TECHNOLOGY (ANATEC) CRISP Programmer´s Manual, Oct. 1981, ANATEC, Dublin, OH.

Brown 80
BROWN, M.E., AND WEIDE, B.W. Preliminary design of a highly parallel architecture for real-time applications, in *Proc. 18th Ann. Allerton Conf. on Comm., Contr., and Comp.*, Univ. of Ill., Oct. 1980, pp. 534-543.

Brown 81
BROWN, M.E., AND WEIDE, B.W. A flexible packet-switching chip set and its applications, in *Proc. 19th Ann. Allerton Conf. on Comm., Contr., and Comp.*, Univ. of Ill., Oct. 1981, pp. 534-541.

Dennis 74
DENNIS, J.B. First version of a data-flow procedure language, in *Lecture Notes in Computer Science 19*, Springer-Verlag, Heidelberg, 1974, pp. 362-376.

Dijkstra 76
DIJKSTRA, E.W. *A Discipline of Programming*, Prentice-Hall, Englewood Cliffs, N.J., 1976.

Hoare 78
HOARE, C.A.R. Communicating sequential processes, *Commun. ACM 21*, 8 (Aug. 1978), 666-677.

Jensen 79
JENSEN, K., KYNG, M., AND MADSEN, O.L. A Petri Net Definition of a System Description Language, Apr. 1979, Comp. Sci. Dept., Aarhus Univ., Aarhus, Denmark.

Kernighan 81
KERNIGHAN, B.W., AND MASHEY, J.R. The UNIX programming environment, Computer 14, 4 (Apr. 1981), 12-24.

Lamport 78
LAMPORT, L. Time, clocks, and the ordering of events in a distributed system, Commun. ACM 21, 7 (Jul. 1978), 558-565.

MacQueen 79
MACQUEEN, D.B. Models for Distributed Computing, Apr. 1979, IRIA report no. 351.

Mashey 76
MASHEY, J.R. Using a command language as a high-level programming language, in Proc. 2nd Intl. Conf. on Software Eng., , Oct. 1976, pp. 169-176.

Milner 77
MILNER, R. Flowgraphs and Flow Algebras, Jul. 1977, Dept. of Comp. Sci., Univ. of Edinburgh, CSR-5-7.

Schutz 79
SCHUTZ, H. A. On the design of a language for programming real-time concurrent processes, IEEE Trans. Softw. Eng. SE-5, 3 (May 1979), 248-255.

Sharp 79
SHARP, J.A. The data flow approach to the design of a language for parallel processing, in Distributed Computing Project Report No. 2, Univ. of Warwick, Coventry, England, Apr. 1979, pp. 55-67.

Wegner 79
WEGNER, P. Programming with ADA: An Introduction by Means of Graduated Examples, Prentice-Hall, Englewood Cliffs, N.J., 1979.

Wirth 77
WIRTH, N. A language for modular multiprogramming, Softw. Pract. and Exper. 2, 1 (Jan.-Feb. 1977), 3-36.

Yamano 81
YAMANO, K., AND YOSHIHARU, M. Unified functional design technique based on data flow concept, in Proc. COMPSAC81, IEEE, Chicago, Nov. 1981.

VARIETIES OF DECOUPLED ACCESS/EXECUTE COMPUTER ARCHITECTURES

JAMES E. SMITH
Department of Electrical and Computer Engineering
University of Wisconsin-Madison, Madison, WI 53706

THOMAS J. KAMINSKI
Nicolet Instrument Corporation, Madison, WI 53711

ABSTRACT

Decoupled Access/Execute Computer architectures achieve high performance by using separate instruction streams for data access and execution. The two loosely coupled streams communicate through hardware queues. This paper discusses several extensions and refinements of decoupled architectures. These extensions include single instruction stream versions where the stream is split into access and execute streams at load or run time, and symmetric architectures where the access and execute processors are identical. Also included are performance estimates and comparisons.

1. INTRODUCTION

Decoupled Access/Execute computer architectures [1-3] separate instruction processing into two parts:

1) access to memory to fetch operands and store results, and
2) functional unit execution to produce the results.

This is accomplished by using two conceptually separate instruction streams (one for access, one for execution) that permit the decoupling at the architectural level.

Two processors, the Access-processor (A-processor) and eXecute-processor (X-processor), each execute one of the instruction streams. Communication between the two processors is via hardware queues. There are queues for passing both data and control information (conditional branch outcomes). Such an architecture has the following advantages:

1) It permits high-speed instruction issue; up to two instructions can be issued per clock period. This is twice the rate for traditional single instruction stream processors which contain the Flynn bottleneck [4], i.e. some bottleneck in the instruction fetch/decode path where instructions can pass at only one per clock period.

2) It permits a constrained type of out-of-order instruction issue which allows data access operations to be done well in advance of when they are needed, but without resorting to complicated methods that have been used in the past for out-of-order issue [5,6].

3) Unpredictable (and long) memory access delays can be smoothed and possibly hidden. This is particularly important in modern multiprocessor systems where memory bandwidth is a critical resource.

Decoupled Access/Execute (or, simply "decoupled") architectures were defined and discussed in [1]. Since that time, several refinements and extensions have been made, and they are the topic of this paper. Section 2 contains an overview of the original decoupled architecture concept. This includes a refinement of queue architecture that results in a savings of an architectural register over the approach used in [1]. Also in Section 2 is a set of performance estimates and comparisons that are made under a more accurate set of assumptions than were used in [1]. Section 3 describes ways two instruction streams can be managed in a decoupled architecture and presents an implementation with a single instruction stream. Performance estimates are given. Section 4 contains a description of a symmetric decoupled architecture where either processor can perform loads and stores for the other processor as well as itself. This approach has several advantages, and they are discussed.

2. DECOUPLED ARCHITECTURE OVERVIEW

Fig. 1 shows a block diagram for a decoupled architecture. The A-processor and X-processor each have their own instruction stream and register file. Since we are dealing with high performance systems, we assume that instruction buffers or caches are available in both processors to reduce instruction fetch traffic. In fact, in later examples we neglect it; a valid assumption for the loop-dominated scientific code we will be considering.

The A-processor performs all operations necessary for transferring data to and from main memory. That is, it does all operand address computation and memory read and write requests. Data fetched from memory are either used internally by the A-processor, or are placed in a FIFO queue going into the X-processor (the X Load Queue or XLQ). The X-processor removes operands from the XLQ as it needs them, uses them for computation, and places results into a second queue, the X-Store Queue, or XSQ.

The A-processor issues memory stores as soon as it computes a store address; it does not wait until store data are in the XSQ. Such store addresses awaiting data are held by the A-processor in the Access Store Address Queue or ASAQ. As a data word becomes available in the XSQ, it is paired with the first address in the ASAQ and both are sent to memory. The pairing takes place automatically as data become available.

A potential problem that might arise when stores are handled as just described, is that a load instruction might use the same memory location (address) as a previously issued, but not completed, store instruction. A solution is to compare each newly issued load address with the members of the ASAQ, holding the load (and all subsequent loads) until all matching conditions go away (i.e. the store are initiated). A slightly more elaborate solution is to "short circuit" the store data to the load path when the store data become available.

In order for the A- and X-processors to work together, they must be able to coordinate conditional jumps or branches. This is also done via queues, the X to A Branch Queue (XABQ) and A to X Branch Queue (AXBQ) as shown in Fig. 1. Either processor could have the data necessary to decide a conditional branch. Consequently, each processor has a set of conditional branch instructions that operate on its own internal registers. There is also a "Branch from Queue" (BFQ) instruction in each processor that is conditional on the branch outcome at the head of the branch queue coming from the opposite processor. When a processor decides a conditional branch outcome, it places the outcome on the tail of the branch queue to the opposite processor. Thus, conditional branches appear in the two processors as complementary pairs. For each conditional branch tested in the A-processor, there is a corresponding BFQ in the X-processor and vice-versa.

Because the XLQ and XSQ are used in much the same way as registers, we use the X0 register designator in instruction operand fields to denote them. When the XLQ is to be used as a source operand, X0 will be used in the appropriate instruction source field. Similarly, X0 will denote the tail of the XLQ when an operand for the X-processor is being loaded by the A-processor. In addition, X0 will also denote the tail of the XSQ when used as a destination by the X-processor, and the head of the XSQ when it is a source in the A-processor. This scheme of architecturally specifying queues as if they are registers is similar to the one proposed in [1], but the method here saves an architectural register since only X0 is used rather than X0 and X1 as in [1].

Examples in this paper are made with respect to a simplified CRAY-1-like scalar architecture. The CRAY-1 architecture [7] was chosen because it:

1) represents the state-of-the-art,
2) already has an instruction set that to some extent decouples access and execution,
3) has a straightforward implementation which makes timing calculations relatively simple.

The following clarifying simplifications to the CRAY-1 architecture were made:

1) Scalar registers are labeled X0, X1, ..., rather than S0, S1, ...
2) Jumps are made with respect to A7 and X7 since A0 and X0 are used to denote queues in the decoupled architecture.
3) There are no memory bank conflicts.

Fig. 2a contains the FORTRAN for the first Lawrence Livermore Loop [8], a set of benchmarks originally intended to measure scalar computer performance. Fig. 2b contains a compilation onto the simplified CRAY-1 architecture. The CRAY FORTRAN compiler with the vectorizer turned off was used as a guide. Fig. 2c contains a compilation onto a decoupled architecture; here the CRAY-1 compilation was again used as a guide, and no further optimization was done.

To the right of the code in Figs. 2b and 2c is the clock period at which the instruction would issue for a typical pass through the loop. CRAY-1 timing was used, except for the assumption of no memory bank conflicts. This simplification in some cases might slightly improve CRAY-1 performance as well as the decoupled architecture performance. After the first few passes through its loop, the A-processor will run far enough ahead of the X-processor so that in the steady state the X-processor never has to wait for memory data. This is assumed for the X-processor timing in Fig. 2c. Including 5 clock periods for the branches terminating the loops, the simplified CRAY-1 takes 40 clock periods per loop pass, while the decoupled architecture takes 20 clock periods per loop pass; there is a speedup of 2.

Table 1, columns 2, 4, and 6, contain a performance comparison for all 14 Lawrence Livermore Loops. These comparisons use CRAY-1-like timing parameters, and are based on fewer simplifying assumptions than were used in [1]. The assumption causing the greatest difference with [1] is that the Flynn bottleneck in the CRAY-1 is actually one instruction parcel (2 bytes) per clock period rather than one instruction per clock period. This means that loads, stores, and branches require a minimum of two clock periods to issue. The one instruction per clock period simplification used in [1] was more beneficial for the CRAY-1 timing estimates. Hence, the more accurate estimates given here show a slightly greater performance improvement than the earlier study.

3. INSTRUCTION STREAM ARCHITECTURE

As described in Section 2, a decoupled architecture uses two separate instruction streams, and an implementation of the architecture has two separate program counters and instruction fetch/decode units. A duplication of instruction fetch/decode units is necessary for beating the one instruction per clock period Flynn bottleneck.

The use of two interacting instruction streams will lead to added software difficulties, however. New techniques for code compilation will probably have to be developed, and a new variety of software errors will be made possible if the queues should be mismanaged. For example, the X-processor might expect one more operand on the XLQ than the A-processor delivers, causing the system to hang up.

An alternative is to define a separate "assembler architecture." It would appear much like a traditional single stream architecture. Code would be written and compiled to adhere to the assembler architecture. Furthermore, correct code from the assembler architecture viewpoint would be free of errors due to queue mismanagement. At program load time, just before execution, the single stream assembler architecture could be split into two streams, with branch from queue instructions inserted at appropriate branch points.

To be more specific, we now define a single stream assembler architecture for the two-stream machine architecture introduced in Section 2. The assembler architecture is very similar to the scalar CRAY-1 architecture except:

1) all loads to X-registers must be to X0 (the queue designator),
2) the only X-register that can be stored is X0 (the queue designator),
3) any time X0 is the destination of a load, it must be used as the source of some other instruction before it is loaded again,
4) any time X0 is the destination of a functional instruction, it must be stored,
5) branch targets are specified symbolically; that is, one does not use actual addresses in the assembler architecture.

Fig. 3 shows a modification of the Fig. 2 code that adheres to the above assembler architecture. The code in Fig. 3 looks like, and can be interpreted in the same way as, a traditional architecture. At load time, the code is be split into two streams. Branch instructions are duplicated; a conditional branch on an X register (A-register) is placed in the X-unit (A-unit) stream with a matching branch from queue instruction being placed in the A-unit (X-unit) stream.

Another alternative is to actually load and execute the single instruction stream program, with splitting into A and X streams occurring at run time as instructions are decoded. Fig. 4 shows a possible implementation for this type of architecture. Instructions are fetched, decoded, and placed into one of two queues, depending on whether they are an X-unit or A-unit instruction. A conditional branch instruction temporarily halts this process while the branch condition is decided by the appropriate unit (A or X). Then, instruction processing continues from the correct instruction address.

One advantage of this method is that hardware costs may be reduced. Only one instruction fetch unit is required, and conditional branch instructions are handled without the need for branch queues. A disadvantage is that the Flynn bottleneck has been re-introduced. That is, the stage where the instruction stream is split handles only one instruction per clock period. This doesn't necessarily lead to a loss of performance, however.

This can be illustrated by the HYDRO EXCERPT example given earlier. The X-program dominates, i.e., the X-program execution time essentially determines the total execution time. Included in the X-unit time is the 5 clock periods for a branch from queue instruction. In a single stream implementation, the branch from queue instructions are not present, and the X-unit execution time can be reduced by a corresponding amount. Fig. 3 shows the single instruction stream version of HYDRO EXCERPT annotated with timing information showing the clock period when an instruction passes the SPLITTER register. This timing indicates that each pass through the loop requires 19 clock periods. This represents the maximum rate allowable due to the Flynn bottleneck. Further analysis shows that in steady state the loop could actually be executed at that rate, since the X-unit instructions now require 15 clock periods. This represents a slight performance improvement, 19 clock periods versus 20 clock periods, and the Flynn bottleneck restricts any further improvement.

Table 1, columns 3 and 4, show the complete timing results for a single instruction stream processor, and a comparison with the two stream processor. There is an improvement for some loops and a loss for others where the Flynn bottleneck becomes a problem.

5. SYMMETRIC PROCESSOR ARCHITECTURE

Thus far we have discussed decoupled architectures in which the A and X processors serve special purposes; they could even have different word lengths. There are, however, significant advantages to making the processors identical.

We first describe the architecture for a symmetric decoupled system. Refer to Fig. 4. Both processors have input queues for load data. Each processor can issue a load instruction to load its own queue or the other processor's queue. Since either processor can load either input queue, some coordination is required to assure determinism. We must also add a new set of conditional branch instructions to each processor's repertoire. These internal branches do not put an outcome on a branch queue for the other processor. This allows the processors to diverge and run separate pieces of code containing the internal branching.

Placing an input queue and output queue in the A-processor leads to the useage of A0 to designate the queue heads and tails in the same way X0 was used in the X-processor. This means only A0 is loaded from memory or stored to memory.

Apart from allowing the use of the same processor design for both the A- and X-processors, this has other advantages. First, it effectively reduces the number of registers required for intermediate results. For example, Fig. 5a shows code for a loop with a conventional A-unit architecture. Fig. 5b shows the same loop with the A unit input and output queues. In Fig. 5a a total of 4 registers are needed for well-scheduled code. In Fig. 5b only 2 registers and the queues are needed.

Second, it makes implementation of the register file simpler. In a CDC/CRAY-type pipelined scalar architecture, there is a single result bus into the register file from the functional units. Data from the memory load path, may require a second bus into the registers -- creating the need for a two write per clock period register file. Or, one can use the functional unit result bus for load data as well. This is done in the CRAY-1 and causes a bottleneck because at most one register can be loaded during any clock period. By using queues, there is a separate load path feeding into the queue rather than the register file. Hence, it is not necessary to write two registers, and the load path still has its own bus.

Another advantage of this method is that it provides more software flexibility. The processors can run in decoupled mode when it improves performance, but there are some problems where the data/address dependencies are such that best performance can be achieved if the A-processor does all the computation and the X-processor sits idle. In such instances the processors can enter a "free-running" mode where they each execute different processes, do their own loads and stores and execute internal branches only. That is, they operate just like a traditional dual processor system.

The processors can switch between decoupled and free-running modes during execution of a single job. For example, they might start a job in decoupled mode, then split apart and free-run for awhile, and finally merge and run in decoupled mode to finish the job. One problem this presents is that a processor, e.g., the X-processor, is loaded by the A-processor in decoupled mode and by itself in free-running mode. All the loads go into the same queue, however, and to ensure determinism the processors would have to synchronize before changing the system's mode of operation. This can be easily accomplished by placing a pair of branch instructions in each stream at the place where synchronization is to occur. This is illustrated in Fig. 6. The JAM and BFQ instructions are essentially no-ops. However, they do interact via the branch queues. It can be seen that whichever processor reaches the synchronization point first must wait for the other before it can pass through to subsequent instructions.

SUMMARY AND CONCLUSIONS

This paper discussed three principal variations of decoupled architectures:

1) The approach originally proposed in [1]; different queue architectures were discussed, and more accurate performance figures were given.
2) Single instruction stream architectures; versions where splitting of instructions into A and X streams occur at load time and run time were both discussed. Performance figures were given that show performance may not be signficantly affected if one instruction fetch unit is used and stream splitting occurs at run time.
3) Symmetric architectures; it was shown that identical procesors can be used for both A and X instruction streams. This allows more software flexibility because it is possible to run in both a decoupled and a free-running mode.

A conclusion that can be drawn from this paper is that high performance can be achieved by decoupling operand access and execution. We feel, however, that a more important conclusion is that there are many varieties of decoupled architectures, some of which were discussed here, and some of which remain to be discovered. They all deserve further in-depth study, and promise to be a fruitful area of research for some time to come.

Table. 1. Performance Estimates for Decoupled Architecture and CRAY-1 (all times are in clock periods)

Loop	CRAY-1	Decoupled		Speedup	
		One-Stream	Two-Stream	One Stream	Two Stream
1	40	19	20	2.1	2.0
2	60	38	29	1.6	2.1
3	28	13	13	2.2	2.2
4	33	16	14	2.1	2.1
5	60	35	38	1.7	1.6
6	64	36	39	1.8	1.6
7	74	56	61	1.3	1.2
8	205	110	113	1.9	1.8
9	102	61	60	1.7	1.7
10	94	76	60	1.2	1.6
11	27	19	19	1.4	1.4
12	27	14	13	1.9	2.1
13	138	114	114	1.2	1.2
14	148	111	111	1.3	1.3

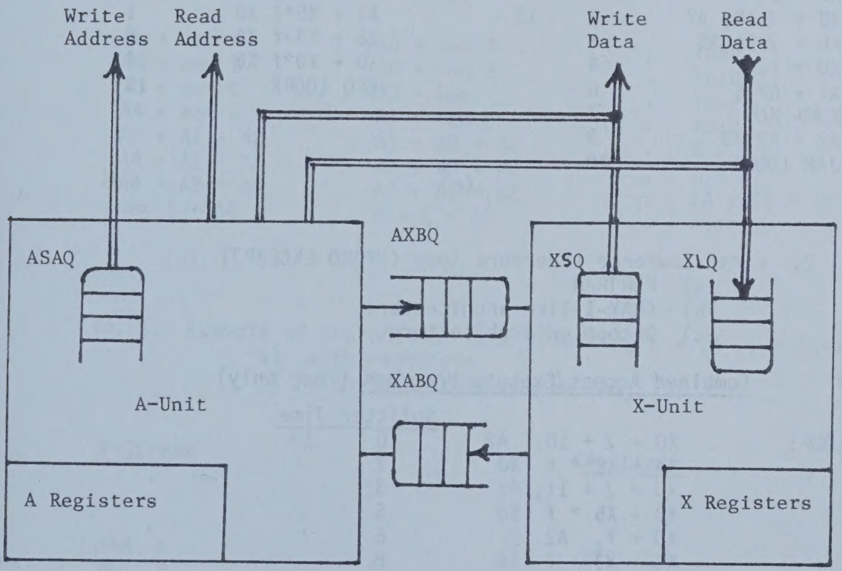

Write Address
Read Address
Write Data
Read Data
ASAQ
AXBQ
XSQ
XLQ
XABQ
A-Unit
X-Unit
A Registers
X Registers

Figure 1. A two instruction stream decoupled architecture.

```
      Q = 0.0
      Do 1  K=1, 400
1     X(k) = Q + Y(k) * (R * Z(K + 10) + T * Z(K + 11))
```

(a)

```
        A7 ← -400          . -loop count
        A2 ← 0             . init. index
        A3 ← 1             . index increment
        X2 ← R             . load loop
        X5 ← T             . invariants          issue time (CP)
Loop:   X3 ← Z + 10, A2    . load Z(K + 10)           0    40
        X7 ← Z + 11, A2    . load Z(K + 11)           2
        X4 ← X2 + f X3     . R * Z(K + 10)           11
        X3 ← X5 * f X7     . T * Z(K + 11)           12
        X7 ← Y, A2         . load Y(K)               13
        X6 ← X3 + f X4     . R * Z( ) + T * Z( )     19
        X4 ← X7 * f X6     . Y(K) * (  )             25
        A7 ← A7 + 1        . inc. loop ctr.          26
        X, A2 ← X4         . store X(K)              32
        A2 ← A2 + A3       . inc. index              34
        JAM Loop           . branch if A7 < 0        35
```

(b)

```
      Access           Issue Time               Execute         Issue Time
        A7 ← -400                             X2 ← R
        A2 ← 0                                X5 ← T
        A3 ← 1                       LOOPX:   X4 ← X2*f X0        0    20
LOOPA:  X0 ← Z+10, A2     0    15             X3 ← X5*f X0        1
        X0 ← Z+11,A2      2                   X6 ← X3+f X4        8
        X0 ← Y,A2         4                   X0 ← X0*f X6       14
        A7 ← A7+1         6                   BFQ LOOPX          15
        X,A2← X0          7
        A2 ← A2+A3        9
        JAM LOOPA        10
```

(c)

Fig. 2. First Lawrence Livermore Loop (HYDRO EXCERPT)
a) FORTRAN
b) CRAY-1-like architecture
c) Decoupled architecture

Combined Access/Execute Programs (loop only)

```
                                     Splitter Time
LOOP:          X0 ← Z + 10,  A2          0     19
               X4 ← X2 * f   X0          2
               X0 ← Z + 11, A2           3
               X3 ← X5 * f   X0          5
               X0 ← Y,  A2               6
               X6 ← X3 + f   X4          8
               X0 ← X0 * f   X6          9
               A7 ← A7+1                10
               X, A2 ← X0               11
               A2 ← A2 + A3             13
               JAM LOOP                 14
```

Fig. 3. Single stream assembler architecture version of HYDRO EXCERPT loop.

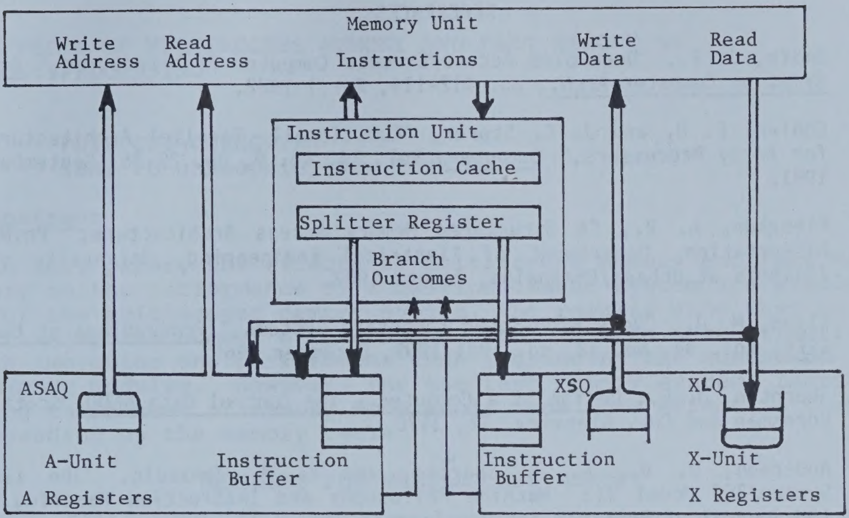

Memory Unit
Write Address
Read Address
Instructions
Write Data
Read Data
Instruction Unit
Instruction Cache
Splitter Register
Branch Outcomes
ASAQ
XSQ
XLQ
A-Unit
Instruction Buffer
Instruction Buffer
X-Unit
A Registers
X Registers

Figure 4. A one instruction stream decoupled architecture.

```
A1 ← mem A          A0 ← mem A          .   load
A2 ← mem B          A0 ← mem B          .   load
A3 ← mem C          A0 ← mem C          .   load
A4 ← mem C          A0 ← mem D          .   load
A2 ← A1 + A2        A1 ← A0 + A0        .   A + B
A4 ← A1 + A3        A2 ← A0 + A0        .   C + D
A4 ← A2 * A4        A0 ← A1 * A2        .   (A + B) * (C + D)
mem E ← A4          mem E ← A0          .   store

    (a)                 (b)
```

Fig. 5. Example of A-processor code computing E ← (A+B) * (C+D)
a) with registers
b) with load/store queues (A0)

```
     X-Stream                             A-stream
         .                                    .
         .                                    .
         .                                    .
     JAM  A                               BFQ  C
A:   BFQ  B                      C:       JAM  D
B:   next instruction            D:       next instruction
         .                                    .
         .                                    .
         .                                    .
```

Fig. 6. Interaction at a synchronization point.

REFERENCES

[1] Smith, J. E., "Decoupled Access/Execute Computer Architectures," *9th Symp. on Computer Arch.*, pp. 112-119, April 1982.

[2] Cohler, E. U. and J. E. Storer, "Functionally Parallel Architecture for Array Processors," *Computer*, Vol. 14, No. 9, pp. 28-36, September 1981.

[3] Pleszkun, A. R., "A Structured Memory Access Architecture," Ph.D. Dissertation, Department of Electrical Engineering, University of Illinois at Urbana/Champaign, August 1982.

[4] Flynn, M. J., "Very High-Speed Computing Systems," *Proceedings of the IEEE*, Vol. 54, No. 12, pp. 1901-1909, December 1966.

[5] Thornton, J. E., *Design of a Computer - The Control Data 6600*, Scott, Foresman and Co., Glenview, IL, 1970.

[6] Anderson, D. W., F. J. Sparacio, and R. M. Tomasulo, "The IBM System/360 Model 91: Machine Philosophy and Instruction Handling," *IBM Journal of Research and Development*, pp. 8-24, January 1967.

[7] *CRAY-1 Computer Systems, Hardware Reference Manual*, Cray Research, Inc., Chippewa Falls, WI, 1979.

[8] McMahon, F. H., "FORTRAN CPU Performance Analysis," Lawrence Livermore Laboratories, 1972.

EFFECTS OF MULTIACCESS MEMORY AND FAST MEMORY ON MULTIPROCESSING

PHIL CHI-CHUNG YEH
IBM, Poughkeepsie, New York

Abstract

In this paper, the effects of multiaccess memory and fast memory on the performance of a multiprocessor system are studied. For the multiaccess memory system, the results show that increasing the number of accessing paths is much more effective in improving the performance than increasing the number of memory modules. However, for the fast memory system, increasing the number of memory modules is much more effective than speeding up the memory cycle.

RESULTS FOR MULTIACCESS MEMORY

An analytical model based on binomial approximation was developed to study the effects of multiaccess memory (1) on system performance. The performance measurement is in terms of the probability of acceptance, P_A, for a typical memory request. It is assumed that the multiprocessor system consists of p synchronous processors and m interleaved memory modules. The processor-memory system is interconnected via a cross-bar switch. All memory modules are identical and each consists of k accessing paths. In addition, the addresses of requests are assumed to be independent and uniformly distributed among the m memory modules. Furthermore, the processor request rate is designated to be r and all rejected requests are resubmitted as new requests at the next processor cycle until they are accepted. In the model, the effects on performance of simultaneous read and write accesses or multiple write accesses to the same memory location are neglected.

Some results obtained from the analytical solution for certain reasonable combinations of parameters are summarized as follows:

1. For a given p and k, the same performance, P_A, occurs at constant km/rp.
2. For a given r and km/rp:
   if m is a constant, larger p results in higher P_A;
   if k is a constant, larger p results in lower P_A;
   if p is a constant, larger k results in higher P_A.
3. To obtain a reasonable performance, km/rp is required to be at least one.
4. Increasing k results in a most significant improvement of P_A at km/rp $\approx 1/2$.

RESULTS FOR FAST MEMORY

It is conceivable that the random access memory module may have a cycle time shorter than the processor cycle time in a multiple microprocessor system. Li (2) made a Markov Chain model to study the average memory bandwidth of such a system. However, no general solution exists in his model. A simple probabilistic model has been developed from which general solutions can be obtained. In this model, the following assumptions are made. The processor cycle is n times the memory cycle, where n is an integer. Each processor issues memory requests asynchronously with respect to other processors, so that all rejected requests will be resubmitted as new requests at the next memory cycle. Again, as in the case of multi-access memory model, the request pattern is independent and uniformly distributed among all memory modules and the processor request rate is given by r.

Some results for such a p x m multiprocessor system are summarized as follows:

1. For a given p and m, the same performance occurs at constant nm/rp.
2. For a given r and nm/rp:
   if m is a constant, larger p results in lower P_A;
   if n is a constant, larger p results in lower P_A;
   if p is a constant, larger m results in higher P_A.
3. To obtain a reasonable performance, nm/rp is required to be at least 2.
4. Increasing n results in a most significant improvement of P_A at nm/rp $\approx$ 1.

References

(1) W. W. Chu and P. B. Korff, "Multiaccess Memory: An Overview," Proc. 4th Texas Conf. Computing Syst., Austin, Tx., 1975

(2) H. F. Li, "Bandwidth of Fast Memory in Multiprocessing," Proc. IEEE, Vol. 68, No. 5, pp 630-632, May 1980

A VLSI-BUILDING BLOCK FOR THE PROCESSOR-TO-MEMORY COMMUNICATION IN LARGE-SCALE MULTIPROCESSOR SYSTEMS*

B. QUATEMBER
University of Innsbruck
Innsbruck, Austria

ABSTRACT

In large-scale multiprocessor systems, the implementation of an appropriate interconnection network is still a crucial issue. The paper refers to a new design approach for the extraordinarily well suited full crossbar. This concept is based on the use of entities with considerable intelligence that are spread over the whole crossbar. We call them configurational processors. A detailed description of the functional goals and the structure of such a configurational processor is given. The paper focusses then on the realization of a combination of several configurational processors together with the associated switches in the crosspoints as a single VLSI-building-block.

INTRODUCTION

It is an interesting fact that in the first commonly accepted taxonomy of multiprocessor-systems given by Enslow in 1974 /1/, these systems are classified according to their interconnection scheme. This shows that the interconnection network for processor-to-memory communication was then already a crucial issue in the field of multiprocessor systems.

Since that time, advances in semiconductor technology have made the processors and memory products required for the processing elements and for the shared memory less expensive than the hardware components which are needed for the interconnection network between these processing elements and the shared memory (shared memory modules). This trend will accelerate in the future. The hardware costs of the processors and memory products no longer hinder us to realize large-scale multiprocessor systems. However, in large-scale multiprocessor systems with a wide range of applicablity, high-performance interconnection networks for the processor-to-memory communication are needed, and one has to realize that such a processor-to-memory switch can rapidly become the dominant factor in the cost of the overall system. This is particularly true for a full crossbar processor-to-memory switch /2-8/ which would no doubt be extraordinarily well suited for large-scale systems. Unfortunately, when conventional architectural concepts are employed, large-scale crossbars cannot be implemented with a tolerable amount of hardware, as has been thoroughly investigated in /6/. The main reason for this reality is the poor LSI-compatibility inherent in these concepts. Thus, the gigantic progress in LSI- (VLSI-) technology cannot be fully exploited.

The contemporary state of engineering in the field of multiprocessor systems may be characterized by the following observations:

---

* This work was supported in part by the Austrian Research Councicil, under Grant No. 3896.

1. The problem of the implementation of a large-scale crossbar switch - which is, beyond a doubt, the best interconnection network for large-scale systems - has not yet been solved satisfactorily.

2. The only operative large-scale multiprocessor system which can be regarded as efficient, namely the Cm*-system, is based on a hierarchical interconnection scheme that is less powerful than the crossbar scheme; the inferior quality of the Cm*-interconnection-system is mainly caused by its blocking behaviour (the full crossbar, in contrast, is a non-blocking interconnection network).

There are numerous potential fields of application of large-scale multiprocessor systems that are not appropriate to the hierarchical scheme of Cm*. Consequently, there is a real need for a more powerful and generally applicable large-scale interconnection system which should preferably be a crossbar. Hence, all efforts towards the realization of a large-scale full crossbar are completely justified.

In Reference 7, a new VLSI-compatible approach for the design of a large-scale full crossbar based on so-called "configurational processors" is presented; it will be the basis of our VLSI-building-block.

OVERVIEW OF THE NEW DESIGN APPROACH FOR A LARGE-SCALE CROSSBAR

It is the purpose of the new design approach /6-8/ to introduce an alternative to more conventional crossbar organizations. In this alternative concept, which is particularly suitable in the case of large-scale designs, no unfavourable conditions for the implementation of a large-scale full crossbar arise.

As we know, such unfavourable conditions do arise in conventional crossbar designs (as e.g. in those of the C.mmp-system /2/) whenever the size of the crossbar that has to be built expands from the relatively small size of most of the contemporary systems to the considerable size typical for large-scale systems.

The difficulties are in particular:

1. The enormous hardware complexity caused by the mostly used bit-parallel buses in the data paths.

2. The gigantic amount of hardware for the requisite interconnection structure for the control of the switches in the crosspoints. This extraordinarily high expense is mainly caused by the procedure of the configuration of the crossbar. In a C.mmp-like system, all functional activities in the course of a reference to the shared memory are evoked and completely controlled by the processing element concerned, resulting in long and expensive interconnection lines between processing elements and switches in the crosspoints (Complexity: $O(n.\log n)$ or in special cases $O(n^2)$).

These difficulties have been carefully examined in /6/ to the effect that the only area in which improvements relating thereto are conceivable is the field of architecture.

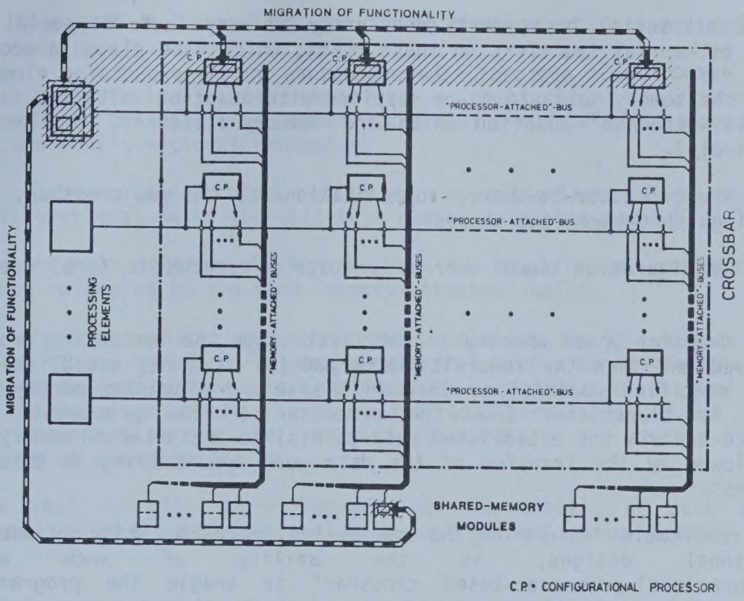

MIGRATION OF FUNCTIONALITY
C.P
"PROCESSOR-ATTACHED"-BUS
CROSSBAR
MIGRATION OF FUNCTIONALITY
PROCESSING ELEMENTS
"MEMORY-ATTACHED"-BUSES
SHARED-MEMORY MODULES
C.P • CONFIGURATIONAL PROCESSOR

Figure 1 - Organization of the new crossbar design

The new design approach /7/ aims at such architectural improvements to overcome the above mentioned difficulties; the approach holds some promise for obtaining a viable organization for future large-scale crossbar switches.

The principle measures /7/ to achieve these central aims of the new concept are:

1. The utilization of bit-serial buses (there are bit-serial "processor-attached" buses and bit-serial "memory-attached" buses).

2. The introduction of entities with considerable intelligence which are distributed among the whole crossbar. We call them "configurational processors". They are intended for the control of the switches in a combination of several adjacent crosspoints (e.g. 16 adjacent crosspoints) on the respective "processor-attached" bus.

3. The exploitation of the "processor-attached" buses to transfer the configurational information to the configurational processors, so that no separate wiring has to be provided for that purpose.

The logic structure of a crossbar based on this new rationale /7/ is represented in Figure 1. For comparison purposes here, an arrangement of the processing elements and the shared-memory modules relative to the crossbar has been chosen which is still in conformity with the analogous arrangement in the C.mmp-system. Later, however, we will use another scheme for that arrangement.

The bit-serial buses must necessarily be very fast bit-serial buses. For the bit-serial transfers on these buses, which take place in accordance with an asynchronous protocol, the processors of the processing elements as well as the memory subsystems are supplied with adaption units (we call them "processor-attached" adaption units and "memory-attached" adaption units, respectively).

In the processor-to-memory communication via the new crossbar, we have to distinguish between two phases:

a. the configuration phase whereby a route or routes is (are) specified, and

b. the transfer phase whereby in the first stage the contention problem is solved and then the requisite data path(s) is (are) established along the specified route(s) so that in the second stage the memory address can be transmitted from the processor of the processing element concerned via the established data path(s) to the attached memory module followed by the transfer of the data word appertaining to this memory access.

A remarkable feature of the new design approach, which is unknown in conventional designs, is the ability of such a new "configurational-processor-based crossbar" to enable the programmer to provide writing operations not only to a single shared memory module but also, and simultaneously, to an arbitrary number of shared memory modules. Thus, a processor of a processing element can broadcast a data item (0 broadcast scope 1). Other distinguishing features appertain to the structural advantages of the new crossbar design, especially to its modularity, resulting in modest hardware-expense even in the case of large-scale systems.

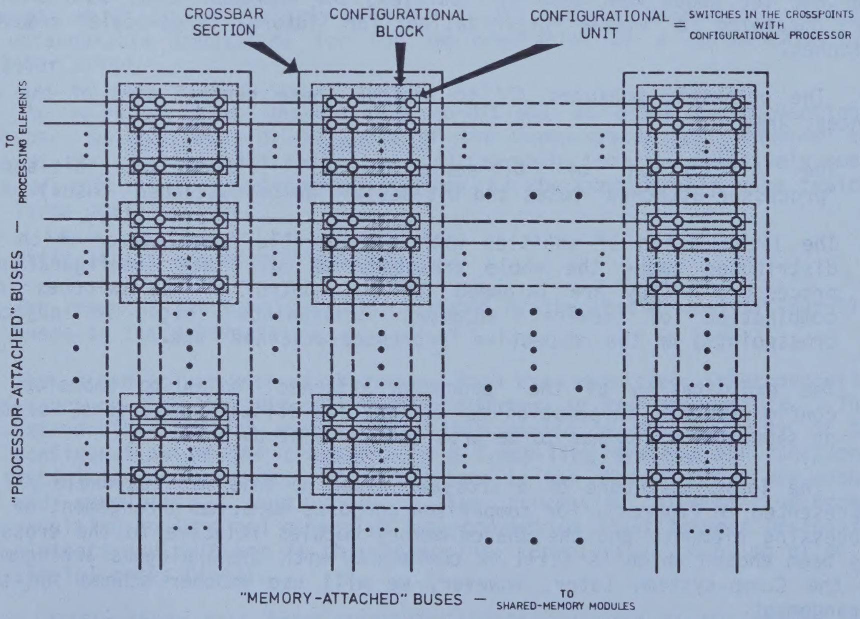

CROSSBAR SECTION
CONFIGURATIONAL BLOCK
CONFIGURATIONAL UNIT = SWITCHES IN THE CROSSPOINTS WITH CONFIGURATIONAL PROCESSOR
"PROCESSOR-ATTACHED" BUSES — TO PROCESSING ELEMENTS
"MEMORY-ATTACHED" BUSES — TO SHARED-MEMORY MODULES

Figure 2 - Canonical crossbar switch structure

In Figure 2, the principal structural elements of this new design are represented, namely

a. the configurational unit which is a combination of a number of crosspoints (we call this number "subdivision-number" / 7/ provided with its own configurational processor,

b. the configurational block which is a combination of a number of configurational units (we call this number "blocking-number" /7/), and

c. the crossbar section which contains the totality of configurational blocks belonging to the same "memory-attached" buses.

In order to achieve an advantageous solution for the physical configuration of the crossbar, (a) one should implement a configurational block as a single building-block, preferably as a VLSI-building-block, and (b) one should endeavour to house (package) the configurational blocks of each crossbar section in very close proximity, thus keeping the expense of wiring and cabling at a minimum.

The rest of this paper focusses on the details of such a VLSI-implementation of a configurational block and on performance aspects.

GENERAL VLSI - ASPECTS

VLSI-design issues

The architectural potential of VLSI stands in sharp contrast to the limited capacities of more conventional design approaches based on SSI-, MSI- and traditional LSI-circuits. It is now possible to produce chips which exceed in complexity of functional elements anything conceived in the past. Hence, it no longer makes sense to persist in minimizing functional elements and keeping them busy all the time.

However, other design issues like (a) the constraints due to the limited number of pins of an IC-package, (b) the amount and expense of "off-chip" interconnection lines, and (c) the structural regularity of the chips which have seldom been crucial in more conventional design approaches in the past now become dominant design aspects in the case of VLSI.

Architectural concepts for VLSI

In particular, the VLSI-technology calls for architectural concepts which allow a subdivision of the whole complex system into IC-realizable subsystems of the greatest possible complexity, whereby the number of input/output/power supply - connections of these subsystems must not exceed the pin-count of an IC-package (maximum: about 60-90 pins). In order to take full advantage of the capacities of VLSI, it is, moreover, essential that these subsystems also exhibit a largely regular structure of their functional elements so that the "on-chip" interconnection topology becomes simple and easy to implement. It is also highly desirable that this subdivision in subsystems (ICs) meet the requirement of a simple and cheap "off-chip" interconnection structure.

VLSI-compatibility of the new design

The question naturally arises whether the new crossbar scheme is a viable concept for a VLSI-realization. To answer it, we have to base our considerations on the size of the above-mentioned VLSI-building-block. Bearing in mind the contemporary advances of technology, the realization of a "configurational block" (Figure 2) with a "subdivision number" of 16 and a "blocking number" of 16 on the area of a single chip might be a realistic assumption. Such a realization would still comply with the pin-count limitations of standard IC-packages. Utilizing VLSI-building-blocks (ICs) with this size, a full crossbar for a large-scale multiprocessor systems with for example 256 processing elements could be built with 256 of such (identical or almost identical) ICs.

If we take the graphical representation of the canonical crossbar structure of the new design (Figure 2) as a basis, it becomes obvious that the "off-chip" interconnection structure will be very simple.

The VLSI-problem of the high degree of structural regularity of the chips is greatly affected by the architectural concept of the configurational processors. An advantageous characteristics of this concept is its massive use of microcode (nanocode) and sections with programmable logic.

ARCHITECTURAL CONCEPT OF A VLSI-REALIZABLE CONFIGURATIONAL BLOCK

General Design Considerations

The new design approach of the full crossbar was introduced for the implementation of the interconnection structure in multiprocessor systems, although it can be exploited in other MIMD-architectures as well and even in SIMD-computer systems /5/. In each of these just-mentioned classes of systems a large variety of types of interconnected components and of methods of their organization is always encompassed.

Restrictions. This wide range of applicability and serviceability naturally entails a considerable diversity of communication schemes resulting in dissimilar requirements for the individual implementation of the configurational processor and so of the configurational unit and of the configurational block. These dissimilar demands will lead to a great variety of architectural approaches. As it would be too complex to treat this topic exhaustively in this paper, we have confined ourselves to an architectural conception of a configurational block which is intended for more specific instances of its employment.

In particular, we have limited our design considerations in the following important ways:

1. We are concerned solely with full crossbars in multiprocessor systems.

2. We further restrict ourselves to the processor-to-memory communication in such multiprocessor systems.

3. Our next restrictions relate to the organization scheme of the existing multiple memory modules, especially to their arrangement and their

facilities: In the system under consideration, every memory module is accessible by every processor. Hence, no private memory exists. However, all of the memory modules are distributed among the whole number of processors in such a manner that each processor is supplied with at least one (or preferably two) of them. Thus, every processor is provided with its own local memory. In addition, our considerations are limited to a further characteristic feature of the organization scheme, namely the direct access of every processor to its own local memory, whereas the other processors may refer to this local memory only via the crossbar.

4. Our last-quoted item deals with restrictions on the character of the operations in the course of a processor-to-memory communication. In the multiprocessor system under consideration, we assume as a fact that the memory references via the crossbar are confined to writing operations whereas reading operations are admissible only in those cases in which a processor refers to its own local memory directly.

Architecture and Implementation Details of the VLSI-Building- Block

Our treatment is based on the assumption that a fixed number of adjacent configurational units (configurational processors + switches in the crosspoints) pertaining to the same "memory-attached" buses is realized as a single VLSI-building-block. In the new concept, the configurational processor is the principal constituent of the architecture of the VLSI-building-block. The only other functional elements of the VLSI-building-block, namely the switches in the crosspoints, can because of their simplicity practically be regarded as belonging to the chip-area of the configurational processor and will therefore not be treated separately in this discussion of the VLSI-implementation.

Partitioning of the configurational processor into a control unit and an execution unit (VLSI-aspects). As in the design of any one digital system, in our case of the design of a configurational processor we must also (a) specify on the one hand a control algorithm for the functional activities, and (b) on the other hand specify the operations which will be controlled by this algorithm. Thus, we must (a) introduce an appropriately organized control unit for the hardware realization of the control algorithm, and (b) provide another unit for the execution of the operations dictated by this algorithm (we will call this unit "execution unit").

In our concept, we use a control unit which is microprogrammed and nanoprogrammed. The control unit is hierarchically structured; it consists of a microprogrammed main control section and several nanoprogrammed subsections. It is thus possible to design this control unit in an orderly and uniform way. In these microprogrammed and nanoprogrammed control sections, the majority of functional elements will be memory cells. The circuitry and especially its (intra-chip) interconnection structure will thus be very regular. Such circuitry can be easily implemented and packaged in high densities on a VLSI-chip.

The above-described partitioning of the configurational processor into a control unit and an execution unit (controlled architectural complex) is to some extent arbitrary. In particular, we can (a) endeavour to specify more elementary operations to be controlled by the control unit, resulting in a simpler execution unit but requiring a more complex control unit, or (b) content ourselves with a less complex control unit but resign ourselves to a sophisticated execution unit. In every design and in every implementation, it is very important that such a partition be achieved in an appropriate manner. This is particularly true for the design of our VLSI-building-block. Because in our case of an intended VLSI-realization favourable conditions for the implementation of the control unit exist, we will, of course, endeavour to subdivide the configurational processor in such a manner that a relatively simple execution unit is attained by using a more complex and sophisticated control unit. We will discuss particular design trade-offs of this kind in more detail later.

A description of the architectural details of the microprogrammed/nanoprogrammed control unit would be beyond the scope of this paper; it will be published in a sepatarate article later. Moreover the length-limit imposed on this paper do not even permit an in-depth discussion of the execution unit; we will resign ourselves to an overview of this unit and a supplementary remark on an important VLSI-detail.

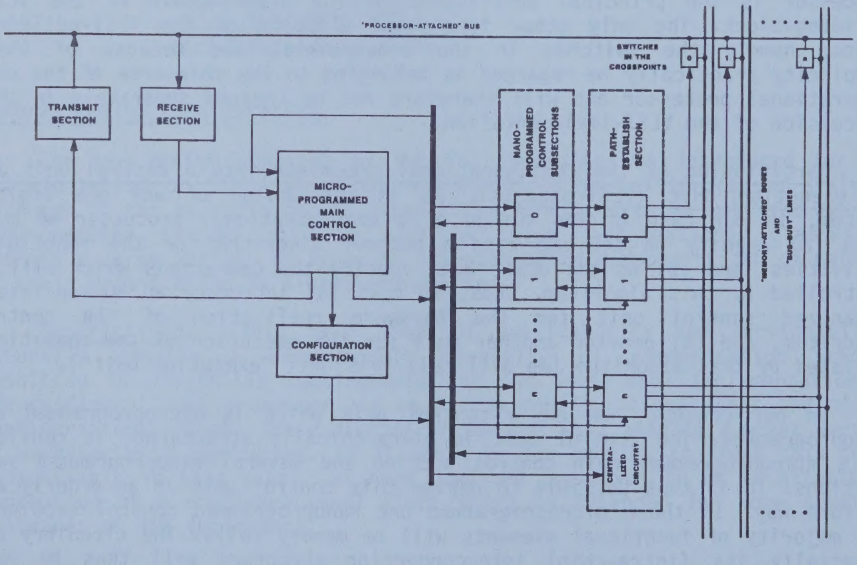

"PROCESSOR-ATTACHED" BUS
SWITCHES IN THE CROSSPOINTS
0
1
n
TRANSMIT SECTION
RECEIVE SECTION
NANO-PROGRAMMED CONTROL SUBSECTIONS
PATH-ESTABLISH SECTION
MICRO-PROGRAMMED MAIN CONTROL SECTION
"MEMORY-ATTACHED" BUSES AND "BUS-BUSY" LINES
CONFIGURATION SECTION
CENTRALIZED CIRCUITRY

Figure 3 - Block diagram of the execution unit

Structure of the execution unit. Our attempts to find an optimal method for this subdivision resulted in the overall structure of the configurational processor shown in Figure 3. As seen in Figure 3, the execution unit is subdivided again, in accordance with the above-mentioned classification of the functional activities in the configurational processor, into the four main sections (sub-units) which follow.

1. The receive section: Its intended use is to capture the information contained in the bit-serial data streams transmitted via the adjoined "processor-attached" bus.
2. The transmit section: It permits transmissions of response control information from the configurational processor to the adaption unit of the processing element concerned.
3. The configuration section: It accepts the items of configurational information and specifies in its configurational section, according to this information, a route (planned path) or a group of divergent routes in the crossbar sections concerned.
4. The path-establish section: It is composed of several more elementary units; there are several elementary units that have an effect solely on one "memory-attached" bus (each and every "memory-attached" bus is provided with such a unit) and there are also other elementary units that are not distributed in such a manner among the "memory-attached" buses. In this section, the required operations for the resolution of the contention problem are executed in a decentralized manner. This resolution is based on different levels of priority, and in the case of contending requests on the same level of priority, on a round-robin scheme.

The speed of the bit-serial data transmissions - a critical VLSI-design issue. It is a fundamental rule that bit-serial data transmissions are slower than bit-parallel data transmissions. In our design approach, bit-serial buses are employed for the data transmissions across the crossbar switch. A straightforward application of this rule may give the impression that a multiprocessor system with such a crossbar as processor-to-memory switch is subject to a performance degradation. This aspect of data transmission is, beyond a doubt, a very critical design issue. In our design approach, however, several measures are provided to prevent possible serious consequences caused by the bit-serial mode of transfer. First of all, however, we have to take a closer look at the above comparison of the two transfer modes. The above statement, "bit-serial transmissions are slower than bit-parallel transfers", can, of course, be made in this straightforward manner only if we take comparable physical realizations of circuitry for granted. Since in our case the bit-serial transmissions obviously will be extremely time-critical, we would employ an ultra-fast technology for the implementation of the circuitry directly involved in these transmissions. It certainly would be advantageous for the performance to endeavour to implement all other parts of the system in a very fast technology as well. However, we have to bear in mind that fundamental limitations are then going to come into play. As the circuit density in our intended VLSI-implementation of a configurational block will be very high, the power dissipation limit of the IC-package and the power dissipation limits in the other parts of the construction will impose strict constraints on the tolerable power consumption of the functional elements (gates) and thus on the attainable circuit speed. It then would be a favourable strategy (a) not to hesitate to exploit an ultra-fast and thus extremely power-consuming version of ECL-technology to realize the relatively simple time-critical sections on the chip involved in the bit-serial data transmissions, and (b) to content oneself with a less power-consuming implementation (slower version of ECL-technology) of the functional elements in the other by far more complex sections of the chip; more specifically, to content oneself with an implementation at such a power level that the limit for the overall power dissipation of the chip is not exceeded, resulting in circuitry with a more limited speed. The ultra-fast circuitry for the data transmission will grant extraordinarily high transfer rates (Baud-rates) so that the efficiency of the bit-serial transfer mode will then be in proportion to the the more limited performance capacities of the other functional units of the chip (especially microprogrammed/nanoprogrammed control sections) due to the more limited speed of these architectural complexes. Thus, we are able to prevent a possible performance degradation as a consequence of bit-serial buses.

REFERENCES

1. Enslow, P.H.,Jr. Multiprocessors and Parallel Processing. New York: John Wiley & Sons, Inc., 1974.

2. Wulf, W.A., and C.G.Bell. "C.mmp - A multi- mini-processor." AFIPS Proceedings of the Fall Joint Computer Conference (Vol.41), Part-II, 1972, pp.765-777.

3. Buchberger,B., and K.Aspetsberger. "A universal variable-topology Multi-Microprocessor- System." Proceedings of the 6th International Symposium on Mini- and Microcomputers and their Applications MIMI '80, Budapest, 1980, pp.136-140.

4. Franklin,M.A., S.A.Kahn, and M.J.Stucki. "Design Issues in the Development of a Modular Multiprocessor Communication Network." Proceedings of the 6th Annual Symposium on Computer Architecture, April 1979, pp.182-187.

5. Albrecht,R. "Concept of a multi-processor parallel processing unit." Computing (Vol. 25), 1980, pp.1-16.

6. Quatember,B. "A Hardware Realization of a variable-topology Multi-Microprocessor System - The Implementation Principles." Proceedings of the 6th International Symposium on Mini- and Microcomputers and their Applications MIMI '80, Budapest, 1980, pp.141-147.

7. Quatember,B. "Modular crossbar switch for large-scale multiprocessor systems - structure and implementation." AFIPS Proceedings of the National Computer Conference (Vol.50), 1981, pp.125-135.

8. Quatember,B. "The Contention Problem in Large-Scale Multiprocessor Systems with a Configurational-Processor-Based Crossbar as Processor-to-Memory Switch." Proceedings of the 19th International Symposium on Mini- and Microcomputers and their Applications MIMI '82, Paris, 1982.

MEASURING PARALLELIZABILITY DEGREE OF PROGRAMS

K. ČULÍK and I.A. NATOUR
Department of Computer Science, Wayne State University
Detroit, Michigan 48202

Abstract: The parallelizability degree of a program is the smallest number of processors needed for achieving the maximal speedup. Some experimental and approximate results are presented when randomly selected sequential programs are executed synchronously.

1. Motivation

Originally, parallel computers (multiprocessors) were special purpose machines designed to speed up computations with special data structures (large arrays). The numbers of processors, Q, were in tens (e.g. 64, 16, etc.).

During the last ten years many different computer architectures and networks were proposed, all of them as general purpose machines (e.g. data-driven [Denn 75,79],[Gost 79], demand-driven [Kell 79], microprocessor network [Magó 79], Cm* [GeJS 82], supercomputer [Swar 82], etc.), which raises the problem of efficiency (the utilization of processors) very urgently, because some experts admit $Q=10^4$, or even $Q=10^6$ processors being available in one multiprocessor.

Thinking about unparallelizable programs (e.g. Horner's rule [Kuck 77], or usual recursive procedures), for which the efficiency would be rediculously low, there seems to be only one solution, namely, multiprogramming. That is, one admits to execute many programs simultaneously to avoid many processors being idle. The assumption is that each program would be parallelized as much as possible to speed it up using as much processors as necessary.

For a given program, Prg, the parallelizability degree of Prg, Pd(Prg) is the number of processors required a) to achieve the maximal possible speedup (or the minimum possible execution time), and b) to get the maximal efficiency.

The efficiency problem of processor allocation in general purpose parallel computers can be solved by a parallel operating system, POS, [Schm 80], in a very similar way as the efficiency problem of storage allocation is solved by usual OS in recent serial computers. The POS will assume each Prg is provided with an integer q, $1 \le q \le Q$, representing the number of processors needed to maximize its speedup, thus $q \approx$ Pd(Prg).

The problem discussed here concerns the determination of approximated value of Pd(Prg) for rather special programs based on some random samples. The experimental results help to clarify which parallelizability degrees one may expect in practical programs.

It is not the goal of this paper to find the best parallelization, although some theoretical insight to get Pd(Prg) is needed, and the necessary definitions and assertions are presented in Sects.2-4. Sects. 5 and 6 present several measurings of approximate Pd(Prg), different approximate speedups, etc.

2. Definitions and Notations

A simplifying assumption "each of the arithmetic operations takes one unit of time" was adopted in [Kuck 77] when the speedup, S_p, and the efficiency, $E_p=S_p/p$, were based on the time T_p being "the number of unit time steps required to perform some calculation using $p \ge 1$ processors."

Being more realistic and general a set, Base, of instructions (assignments and tests) is assumed and with each instruction $in \in$ Base its execution time $eT(in)>0$ is associated. Further, all processors are capable of executing any $in \in$ Base (MIMD machines). Each Prg contains only instructions from Base.

Loop-free programs, Prg, can be considered, because there is only finite number $M\geq 1$ of execution sequences, $es_i=(in_1,in_2,\dots,in_{ni})$, $1\leq i\leq M$; the execution time of es_i is

$$eT(es_i)=\sum_{j=1}^{ni} eT(in_j). \tag{2.1}$$

The execution time of Prg on an usual serial computer(with one single processor)can be defined in several different ways according to the purpose:

(i) $eT_1(\mathrm{Prg})=\max_{1\leq i\leq M} eT(es_i)$; the worst case; (2.2)

(ii) $eT_1^{*}(\mathrm{Prg})=\sum_{i=1}^{M} eT(es_i)/M$; the average;

(iii) $eT_1^{**}(\mathrm{Prg})=\sum_{i=1}^{M}(f(es_i)*eT(es_i))/\sum_{i=1}^{M} f(es_i)$, where a probability distribution of all execution sequences in the form of relative frequencies $f(es_i)$, measured on some representative sample, is assumed.

Further only sequential programs will be discussed, when all instructions are assignments, thus $M=1$; then there is no difference among different approaches in (2.2).

The flow of data is reflected in a data graph of Prg, $DG=\langle Mu,Ed\rangle$, where the set of vertices, Mu, is the multiset of instructions occuring in Prg, and the set of data edges, $Ed\subseteq Mu \times Mu$, represents all data dependencies among instruction occurrences $v,w\in Mu$ as follows:

if $(v,w)\in Ed$ then the instruction occurrence v has to be executed (and completed) before the instruction occurrence w. (2.3)

The DG is a DAG from [AhUl 79], or a permit graph from [Culi 79]. It is necessary to differentiate synchronous and asynchronous execution sequences.

3. Synchronous (parallel) execution sequences

A step of DG is a set $Step=\{v_1,v_2,\dots,v_n\}$ of instructions from Mu such that $n\geq 1$ and

v_i and v_j, $i\neq j$, do not assign to the same variable. (3.1)

The Step is a "n-parallel" assignment introduced in [Mann 74]. Let the number of elements in Step be denoted by $|Step|=n$.

The execution time of Step, expressing the synchronous execution of waiting until the longest instruction is completed, is as follows

$$eT(Step)=\max_{1\leq j\leq n} eT(v_j). \tag{3.2}$$

A synchronous execution sequence of Prg (or DG), ses, is a finite sequence of steps, $ses=(Step_1,Step_2,\dots,Step_k)$, such that the following requirements are satisfied:

(i) $Step_i\neq\emptyset$ for each $1\leq i\leq k$; (3.3)

(ii) $Step_i\cap Step_j=\emptyset$ if $i\neq j$, for all $1\leq i,j\leq k$;

(iii) $\bigcup_{i=1}^{k} Step_i=Mu$;

thus $\{Step_1,\dots,Step_k\}$ is a partition of Mu;

(iv) $(v,w)\in Ed$, $w\in Step_j \Longrightarrow v\in Step_i$ and $1\leq i<j\leq k$.

An $ses=(Step_1,\dots,Step_k)$ is using at most $p\geq 1$ processors if

$$|Step_i|\leq p \text{ for each } 1\leq i\leq k. \tag{3.4}$$

Let $ES_p(\mathrm{Prg})$ be the set of all ses's of Prg using at most p processors. The execution time of an $ses=(Step_1,\dots,Step_k)$ is

$$eT(ses)=\sum_{i=1}^{k} eT(Step_i), \tag{3.5}$$

and the (synchronous) execution time of Prg using p processors is

$$eT_p(\mathrm{Prg})=\min_{ses\in ES_p(\mathrm{Prg})} eT(ses). \tag{3.6}$$

From the previous definitions follows

$$\begin{aligned} &eT_1(\mathrm{Prg})>eT_2(\mathrm{Prg})>\dots>eT_{p_1-1}(\mathrm{Prg})> \\ &eT_{p_1}(\mathrm{Prg})=eT_{p_1+1}(\mathrm{Prg})=\dots=eT_{p_2-1}(\mathrm{Prg})> \\ &\quad\vdots \qquad\qquad\qquad \vdots \qquad\qquad\qquad\qquad \vdots \end{aligned} \tag{3.7}$$

$$eT_{p_{d-1}}(Prg)=eT_{p_{d-1}+1}(Prg)=\ldots=eT_{p_d-1}(Prg)>$$
$$eT_{p_d}(Prg)=eT_{p_d+1}(Prg)=eT_{p_d+2}(Prg)=\ldots$$

where $p_d=Pd(Prg)$ is the parallelizability degree of Prg while $d\geq1$ is called the (parallelism) dispersion, denoted disp(Prg), characterizing the type of step function $eT_i(Prg)$ of integer variable $i=1,2,\ldots,Pd(Prg)$.

As Pd(Prg) is not easy to obtain some approximations of it will be considered. The most natural among them is the number of processors

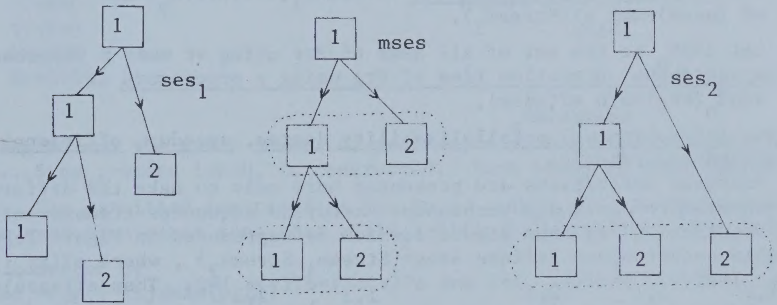
$$M(Prg)=\max_{1\leq i\leq k}|Step_i| \tag{3.8}$$

required when the main ses, $mses=(Step_1,\ldots,Step_k)$, is considered satisfying: each instruction is executed as soon as possible. The inductive definition of mses is as follows:

(i) $Step_1$ contains all instructions $v\in Mu$ such that $idg(v)=0$, that is, no data edge from Ed terminates in v; (3.9)

(ii) if $i\geq1$ and $Step_i$ has been defined already then $Step_{i+1}$ contains all instructions $v\in Mu-\bigcup_{j=1}^{i}Step_j$ such that $idg(v)>0$, and if $(w,v)\in Ed$ then $w\in Step_j$ and $j\leq i$. The construction terminates if $Mu-\bigcup_{j=1}^{i}Step_j=\emptyset$.

Very often $M(Prg)\geq Pd(Prg)$, but it may happen $Pd(Prg)>M(Prg)$ as it is shown is Figure 1.



a) b) c)

Figure 1

In Figure 1 the dotted lines include instructions belonging to the same step, if there are more than one, and integers inside the boxes are execution times of the corresponding instructions. Then $eT(ses_1)=eT_1(Prg)=7>eT(mses)=eT_2(Prg)=5>eT(ses_2)=eT_3(Prg)=4$, disp(Prg)=1, and $M(Prg)=2<3=Pd(Prg)$.

If disp(Prg)>1 then the numbers p_i of processors from (3.7) satisfy

$$1\leq p_1<p_1+1\leq p_2<p_2+1\leq\ldots<p_{d-1}+1\leq p_d=Pd(Prg), \tag{3.10}$$

and will be denoted $a_iPd(Prg)$ for $i=1,\ldots,d$. In particular $a_1Pd(Prg)$ is called the almost parallelizability degree of Prg, the intuitive meaning of which is as follows: $a_1Pd(Prg)$ is the smallest number of processors p such that $eT_1(Prg)>eT_2(Prg)>\ldots>eT_p(Prg)=eT_{p+1}(Prg)$.

Finally one defines the speedup $S_p(Prg)=eT_1(Prg)/eT_p(Prg)$, and reversed inequalities will hold for speedups $S_{p_i}(Prg)$ instead of the execution times $eT_{p_i}(Prg)$ in (3.7).

The efficiencies will satisfy

$$E_p(Prg)>E_{p+1}(Prg)>E_{p+2}(Prg)>\ldots \text{ where } p=Pd(Prg). \tag{3.11}$$

Referring to Prg from Figure 1 one has $S_2(Prg)=7/5<S_3(Prg)=7/4$ and $E_2(Prg)=7/10<E_3(Prg)=7/8$, because of the anomaly.

4. Asynchronous(parallel) execution sequences

We assume $DG=\langle Mu \cup \{\langle dum\rangle\}, Ed\rangle$ of Prg as before with one exception of a dummy instruction, $\langle dum\rangle$, which can be used anywhere, can have any execution time $eT\langle dum\rangle>0$, and has only effect of delaying the execution, as it represents an idle processor.

An asynchronous execution sequence of Prg using $p\geq 1$ processors, ases, is a set of p streams, $Stream_i=(v_{i,1}, v_{i,2}, \dots, v_{i,k_i})$, $1\leq i\leq p$, where $v_{i,j} \in Mu \cup \{\langle dum\rangle\}$ (if $v_{i,j}=\langle dum\rangle$ its $eT(v_{i,j})>0$ is provided) such that

(i) $1\leq j<k_i \Longrightarrow$ either $v_{i,j}\neq\langle dum\rangle$ or $v_{i,j+1}\neq\langle dum\rangle$, and $v_{i,k_i}\neq\langle dum\rangle$; (4.1)

(ii) $v_{i,m}=v_{j,n}$ where $i\neq j \Longrightarrow v_{i,m}=\langle dum\rangle$;

(iii) $v\in Mu \Longrightarrow$ there exists a stream $Stream_i$ and an index j, $1\leq j\leq k_i$, such that $v = v_{i,j}$;

(iv) $(v_{i,;}, v_{j,q})\in Ed \Longrightarrow \sum_{h=1}^{p} eT(v_{i,h}) \leq \sum_{h=1}^{q-1} eT(v_{j,h})$, which means, in the case $i=j$, that $p<q$.

The execution time of a stream $Stream_i$ is

$$eT(Stream_i)=\sum_{a=1}^{k_i} eT(v_{i,h}),\ 1\leq i\leq p. \tag{4.2}$$

The execution time of an ases $=\{Stream_1, \dots, Steam_p\}$ is

$$eT\ (ases)=\max_{1\leq i\leq p} eT(Stream_i). \tag{4.3}$$

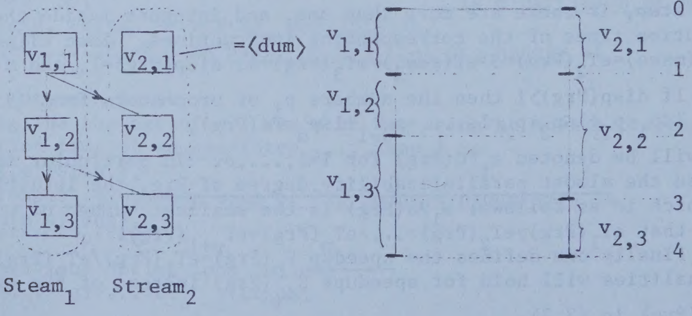
Let $ASES_p$ be the set of all ases of Prg using at most p processors. Then the asynchronous execution time of Prg using p processors is

$$aseT_p(Prg)=\min_{ases\in ASES_p} eT(ases), \tag{4.4}$$

and the (asynchronous) parallelizability degree, speedup, efficiency, can be defined accordingly.

Previous definitions are presented here only to make the difference between synchronous and asynchronous execution sequences transparent.

Considering Prg from Figure 1, which is reproduced in Figure 2a) using a suitable notation, one defines ases=$\{Stream_1, Stream_2\}$, where $eT(v_{1,1})=eT(v_{1,2})=eT(v_{2,1})=eT(v_{2,3})=1$ and $eT(v_{1,3})=eT(v_{2,2})=2$. Then $eT_2(ases)=4=aseT_2(Prg)$, and, obviously $aseT_1(Prg)=7$. Therefore $asS_2(Prg)=7/4$ and $asE_2(Prg)=7/8$. On the other hand $asPd(Prg)=2\neq 3=Pd(Prg)$ which concerns the synchronous execution [2,7].



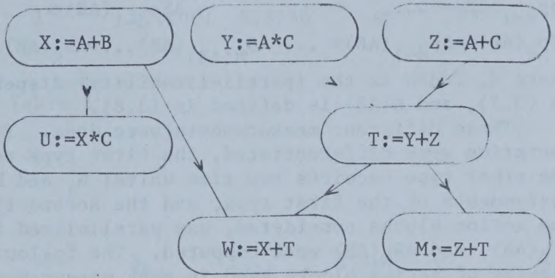
a) b)

Figure 2

The asynchronous case is not discussed further. In particular we are not assuming any particular schedules, etc., as it is the case when OS are investigated.

5. An approximate computation of Speedups

A method to compute an approximate speedup, ASp(AB), of an action block by using p processors is presented. An AB consists of N elementary assignments of the form DEST:=ARG1 op ARG2. Each AB may contain two types of operation, such that the execution time, eT, of one type is greater than of the other. Example 5.1 shows an AB, and its DG is presented in Figure 3.



Example 5.1

X:=A+B
Y:=A*C
Z:=A+C
U:=X*C
T:=Y+Z
W:=X+T
M:=Z+T

Figure 3

Using DG and the given number of processors p, the execution sequence, $Step_1, \ldots, Step_k$, where $1 \le k \le N$, is contructed. Each $Step_i$ is constructed as follows: The permitted instructions $in_1, in_2, \ldots, in_\ell$, are ordered according to their execution times such that $eT(in_i) \ge eT(in_j)$, where $j>i$. The $Step_i$ is then computed by

$$Step_i = \{in_j;\ 1 \le j \le \min(p, \ell)\} \tag{5.1}$$

If p=2 the obtained execution sequence for Example 5.1 is as follows:

$$Step_1 = \{Y{:=}A*X,\ X{:=}A+B\} \tag{5.2}$$

$$Step_2 = \{Z{:=}Z+C,\ U{:=}X*C\}$$

$$Step_3 = \{T{:=}Y+Z\}$$

$$Step_4 = \{W{:=}X+T,\ M{:=}Z+T\}$$

The approximate speedup is computed by

$$AS_p = \sum_{in \in AB} eT(in) \Big/ \sum_{i=1}^{k} \max_{in \in Step_i}(eT(in)). \tag{5.3}$$

By letting eT(*)=2, and eT(+)=1, the approximate speedup for the AB of Example 5.1, and its execution sequence shown above, is 9/6.

The strategy used to determine the execution sequence is not better than some other strategies. In (5.4) there is another execution sequence for AB of Example 5.1 and p=2, which gives an approximate speedup of 9/5. The reason for using the strategy described by (5.1) is because of its simplicity.

$$Step_1 = \{X{:=}A+B,\ Z{:=}A+C\} \tag{5.4}$$

$$Step_2 = \{Y{:=}A*C,\ U{:=}X*C\}$$

$$Step_3 = \{T{:=}Y+Z\}$$

$Step_4=\{W:=X+T,\ M:=Z+T\}$

6. Some simulation results

Because $d(AB)\leq N$ all the time, the approximate speedups could be computed and compared as follows:

$$\begin{array}{ll}
AS_1(AB)<AS_2(AB)<\ldots & \ldots<AS_{p_1-1}(AB)< \\
ASp_1(AB)=AS_{p_1+1}=\ldots & \ldots=AS_{p_2-1}(AB)< \\
ASp_2(AB)=\ldots & \\
\vdots & \vdots \\
ASp_{d-1}(AB)=\ldots. & =ASp_{d-1}(AB)< \\
\multicolumn{2}{l}{ASp_d(AB)=ASp_{d+1}(AB)=\ldots=AS_{M(AB)}(AB)\ldots=AS_N(AB),}
\end{array}$$

where d, $1\leq d\leq N$ is the (parallelizability) dispersion of AB, disp(AB) defined in (3.7), and M(AB) is defined in (3.8).

Three different measurements were done. In all of them two types of operation were differentiated, the first type requires one time unit, while the other type requires two time units; N_1 and N_2 is the number of assignments of the first type, and the second type, respectively. Each of the action blocks considered, was parallelized for p=1,2,...M(AB) and $eT_p(AB)$, and $AS_p(AB)$ were computed. The following is the description of classes of action blocks used in each measurement.

1) Action blocks of N=100 assignments, in which V=20 variables may occur, were generated by a pseudo-random number generator. It was observed that $1\leq disp(AB)\leq 2$, Pd(AB), and aPd(AB) were sufficiently fine concepts, because Pd(AB)-aPd(AB) was small, as shown in Table 1.
2) The same action blocks were used as in 1), except V=75 different variables may occur. As shown in Table 2, it was observed that disp(AB)>2, and Pd(AB)-a d(AB) can be large.
3) Action blocks were randomly selected only from those which satisfies the following requirements:
   i) $DEST_i$ are different for i=1,...,N, and identified with 1,2,...N; which is well known "single assignment" requirement well known in data-flow languages and compilers.
   ii) There is a given set of $M\geq 1$ input variables, INPUT(AB), and $\{ARG1_i,\ ARG2_i\}\subseteq\{1,2,\ldots,i-1\}\cup INPUT(AB)$, for i=1,...,N.

Action blocks used had N=50 assignments, and M=10. As shown in Table 3, it was observed that $0\leq disp(AB)\leq 1$ for all action blocks considered, except one which had disp(AB)=3, and Pd(AB) and aPd(AB) were sufficiently close again.

N_1	N_2	Pd	aPd	S_{Pd}	S_{aPd}
40	60	9	7	5.3333	5.0000
45	55	9	9	5.7407	5.7407
45	55	9	9	5.1667	5.1667
49	51	10	8	5.8077	5.3929
51	49	10	6	4.1389	3.7250
52	48	10	10	4.9333	4.9333
54	46	10	7	4.5625	4.1714
54	46	10	10	4.4242	4.4242
57	43	7	7	4.7667	4.7667
58	42	9	6	4.7333	4.1762

Table 1

N_1	N_2	Pd	$a_1Pd = aPd$	S_{Pd}	S_{aPd}	disp	V
40	60	23	12	14.5455	8.8889	5	74
45	55	29	9	9.6875	5.9615	5	73
45	55	20	12	11.0714	8.6111	3	74
49	51	37	11	10.7857	6.8636	5	74
51	49	16	8	9.3125	5.9600	4	74
52	48	31	13	13.4545	8.7059	5	74
54	46	33	11	12.1667	6.6364	6	75
54	46	25	7	12.1667	5.2143	5	75
57	43	24	8	11.9167	6.2174	5	73
58	42	26	13	12.9091	8.8750	4	75

Table 2

N_1	N_2	Pd	aPd	S_{Pd}	S_{aPd}	disp	V
22	28	10	8	6.5000	5.5714	1	60
23	27	14	10	8.5556	7.0000	1	60
24	36	8	8	4.7000	4.7000	0	59
24	36	9	9	6.3333	6.3333	0	60
26	24	14	8	6.7273	4.9333	3	60
26	24	10	8	6.1667	5.2857	1	60
29	21	10	8	5.9167	5.0714	1	60
29	21	10	10	6.4545	6.4545	0	60
29	21	10	8	5.9167	5.0714	1	60
30	30	10	10	5.0000	5.0000	0	60

Table 3

References

[1] Aho, A., and J. Ullman, Principle of Compiler Design, Addison-Wesley, Reading, MA, 1977, 412-414.

[2] Arjomandi, E., M. J. Fischer, N. A. Lynch, A difference in efficiency between synchronous and asychronous systems, Proceedings of STOC, Milwaukee 1981, 128-132.

[3] Culik, K., Synchronized Parallel Computation and Slowdown of Translator, Lecture Notes in Computer Science 75 (eds. E.K. Blum, M. Paul, S. Takasu), Springer 1979, 67-105.

[4] Culik, K., Almost control-free (indeterministic) parallel computation on permit schemes, Conference Record of the Fifth Annual ACM Symposium on POPL, Tucson, January 1978, 176-184.

[5] Dennis, J. B., and Misunas, D. P., A preliminary Architecture for Basic Data Flow Processor, Proceeding Second Ann. Symp. Computer Arthitecture, January 1975, 126-132.

[6] Dennis, J. B., The Varieties of Data Flow Computers, Proceeding of the First International Conference on Distributed Computing Systems, October 1979, 430-439.

[7] Galil, Z., W. J. Paul, An efficient general purpose parallel computer, Proceedings of STOC, Milwaukee 1981, 247-262.

[8] Gehringer, E. F., A. K. Jones, Z. Z. Segall, The CM* Testbed, Computer, October 1982, 40-53.

[9] Gostelow, K. P., and Thomas, R. E., Performance of a Data Flow Computer, Tech. Report 127a, Dept. Information and Computer Science, University of California, Irvine, October 1979.

[10] Juelich, O., and C. Foulk, Compilation of Acyclic Smooth Programs for Parallel Execution, ACM Trans. Programming Languages and Systems, Vol. 3, No. 1, January 1981, 24-48.

[11] Keller, R. M., S. Patil, and G. Lindstorm, A Loosely Coupled Applicative Multiprocessing System, AFIPS Conference Proceedings, Vol. 48, 1979 NCC, 613-622.

[12] Kuck, D. J., Parallel Processing of Ordinary Programs, in Advances in Computer, Vol. 15, Academic Press, New York, 1976, 119-179.

[13] Kuck, D. J., A Survey of Parallel Machine Organization and Programming, ACM Computing Survey, Vol. 9, 1977, 29-59.

[14] Magó, G. A., A Network of Microprocessors to Execute Reduction Languages, Part I and II, Int. Journal of Comp. and Inf. Sciences, Vol. 8 (1979), 349-385, 435-471.

[15] Manna, Z., Mathematical Theory of Computation, McGraw Hill 1974.

[16] Schmidt, R. A., A Parallel Operating System for MIMD Computer, Proceedings of the 1980 International Conference on Parallel Processing, August 1980, 3-4.

[17] Schwartz, J. T., Ultracomputers, ACM TOPLAS, Vol. 2, October 1980, 484-521.

[18] Stone, H. et al, Introduction to Computer Architecture, SRA, Chicago, 1980, 361-425.

[19] Swartzlander, E. E., and B. K. Gilbert, Supersystems: Technology and Architicture, IEEE Trans. Computers, Vol. C-31, 1982, 399-409.

[20] Wei, M., and H. Sholl, An Expression Model for Extraction and Evaluation of Parallelism in Control Structures, IEEE Trans. on Computers, Vol. C-31, No. 9, 1982, 851-863.

ON RUN-TIME PARALLELISM IN SEQUENTIAL PROGRAMS

G. D. LAKHANI
Department of Electrical Engineering/Computer Science
Texas Tech University
Lubbock, Texas 79409

ABSTRACT

Dependence graphs have been used as an instrument for implementating optimizing compilers and for extraction of parallelism in ordinary programs. We suggest that if the length of dependence arcs (distance between dependent computations) is considered, properties of the execution flow graph of programs can be obtained. These properties can be used in stream-lining computations which then could be run in parallel. This paper is concerned with parallelism of FORTRAN DO loops.

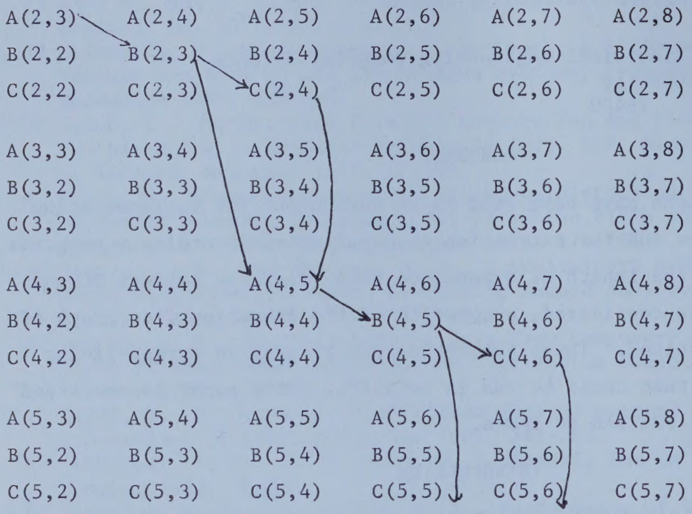
INTRODUCTION

It is not enough to determine simultaneity based only on static structures of sequential programs during program translation. The type of parallelism which is inherent in executions guided by the availability of data (operands) is a simultaneity that can suit for parallel computation on several modesl of processors, especially the dataflow processors and MIMD machines. To extract this type of parallelism, compilers should try to convert a sequential program into a stream of computations. This may require that the compiler study dynamic properties of programs, e.g. scope of variables, side-effects. For this purpose, data dependence and flow dependence should be considered between values and between operations, respectively, rather than between corresponding variables and statements. The example below shows that approach -

Example 1:

```
           DO 2 I = 2, N
           DO 2 J = 2, N
S1:        A(I,J + 1) = B(I-2, J-1) - C(I-2,J)
S2:        B(I,J)     = A(I,J) *2
S3:        C(I,J)     = B(I,J-1) *10
2          CONTINUE
```

A partial execution flowgraph is given below:


A(2,3) A(2,4) A(2,5) A(2,6) A(2,7) A(2,8)
B(2,2) B(2,3) B(2,4) B(2,5) B(2,6) B(2,7)
C(2,2) C(2,3) C(2,4) C(2,5) C(2,6) C(2,7)
A(3,3) A(3,4) A(3,5) A(3,6) A(3,7) A(3,8)
B(3,2) B(3,3) B(3,4) B(3,5) B(3,6) B(3,7)
C(3,2) C(3,3) C(3,4) C(3,5) C(3,6) C(3,7)
A(4,3) A(4,4) A(4,5) A(4,6) A(4,7) A(4,8)
B(4,2) B(4,3) B(4,4) B(4,5) B(4,6) B(4,7)
C(4,2) C(4,3) C(4,4) C(4,5) C(4,6) C(4,7)
A(5,3) A(5,4) A(5,5) A(5,6) A(5,7) A(5,8)
B(5,2) B(5,3) B(5,4) B(5,5) B(5,6) B(5,7)
C(5,2) C(5,3) C(5,4) C(5,5) C(5,6) C(5,7)

The arrows show that the complete flowgraph can be divided into disjoint stream of computations, and each stream consists of computation of A(I,J), B(I,J), C(I,J + 1) repeating with I ← I + 2 and J ← J + 2. Each stream can be run asynchronously. One translation of the loop that describes this type of parallelism is given below.

```
FORALL (I,J) in {2,3} X {2,..,N} U {4,...,N} X {2,3} DO

      IF (I>3 & J = 2) B (I-2,J) - A(I-2, J) *2
      IF (I>3 & J ≥ 2) C (I-2,J) = B(I-2, J-1) *10
      DO 1 K = 0, *, 2 WHILE (I + K ≤ N & J + K + 2 ≤ N)
            A(I+K, J+K+1) = B(I-2+K, J+K-1) + C(I-2+K, J+K)
            B(I+K, J+K+1) = A(I+K, J+K+1) *2
            C(I+K, J+K+2) = B(I+K, J+K+1) *10
1 CONTINUE —
      IF (J+K+2 > N+1) A (I+K, J+K+1) = B(,) + C(,)
      IF (J+K+2 > N) B (I+K, J+K+1) = A(,) *2
END FORALL
      Compute B(N-1, 2), B(N,2) C(N-1, 2), C(N,2)
                 C(N-1, 3), C(N,3)
```

Actually, the WHILE can be dropped by computing the exact value of *, and IF tests can be deleted by doing certain redundent executions. The FORALL is a paraphrase that has been developed in some dataflow language, e.g, VAL, and has been used before [Padu 80].

The paper is concerned with FORTRAN - like DO loops of the form:

```
DO 2 $I_1$ = 0, $u_1$
⋮
DO 2 $I_d$ = 0, $u_d$

    $S_1$ $(\bar{I})$
    ⋮
    $S_N$ $(\bar{I})$

2 CONTINUE
```

Kuck and his associates at the University of Illinois have considered the problem of translation of loops of this form for parallel execution on various types of architectures over many years [Kuck 74, Bane 79, Padu 80]. In a recent paper [Kuck 81], they used the dependence graphs as an instrument for various kinds of transformations which could be used by compilers for optimization of codes and for extraction of vector operations. Their most general algorithm on parallelization of loops is based on the loop distribution principle [Bane 79] which we will summarize soon. We will now define some terms used in this paper.

The index $\bar{I}$ denotes the index vector $(\bar{I}_1,..,\bar{I}_N)$ in the loop. The statements S_r are linear, assignement statements without having any side-effects. Let $\bar{i}$ and $\bar{j}$ be two instances of $\bar{I}$, and let S_r and S_t be any two statements, not necessarily distinct, so that $S_r(\bar{i})$ is executed before $S_t(\bar{j})$ in the sequential execution of the loop. The definition of dependence given in [Bane 79] states that S_r is dependent on S_t if either

(i) the output variable of $S_r(\bar{i})$ is an input variable in $S_t(\bar{j})$

or

(ii) the ouput variable of $S_t(\bar{j})$ is an input invariable in $S_r(\bar{i})$.

The two dependence are called (i) data, (ii) antidata dependence. [Kuck 81] defines some more types of dependence that we do not need here. Notice that the dependence has been defined between statements and not between the values of output or input variables. An algorithm to determine whether or not a statement having an array variable of one dimension is dependent on another which also have the same array is given in [Bane 79].

The dependence graph of a loop is a directed graph, say G, with statements $S_1,\dots,S_N$ as its vertices, and dependence between statements as arcs of G. The vertices of G can be partitioned into strongly connected components (a maximal set of vertices so that there is a path between every two vertices). The statements that belong to a SCC are, essentially, mutally dependent, and therefore need to be executed in the order given by the original loop. The SCCs of G can be divided into levels (so that a SCC in a level depend on at least one SCC in the previous level). The loop distribution principle states that statements in SCCs of one level can be executed in parallel, and execution can flow from one level to the next, sequentially. Therefore, for our purposes, we shall assume that dependence graph of the given loop contains only one SCC.

Statements which do not belong to any cycle in the dependence graph can be separated, and all its iterations can be translated into a vector operation. Cycles which contain dependence arcs which arise due to antidata and output dependence can be avoided. Algorithms which can delete such arcs from the dependence graph are obtained in [Kuck 81]. Statements that remain in some cycle are then executed sequentially. In order to execute statements which belong to cycles, all instances of statements can be translated into linear recurrence equations, and various algorithms then could be applied to solve these equations [Bane 79].

The purpose of this paper is to consider dynamic properties of programs which can describe run-time behavior. Distinction should be made between statements and operations associated with those statements because operations are dynamic and are assoicated with execution of the programs where as statements describe static structure of programs. This is the key observation made by Heuft and Little in [Heuf 82] which they used to show by taking some examples that time-processor bounds calculated on the basis of static analysis of programs could be much higher than the bounds based on the dynamic study of programs. We use their this observation to analyze dependence graphs which then could be used in obtaining translation of loops based on their dynamic behavior.

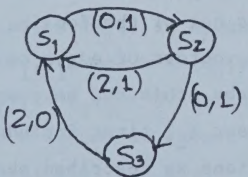
II. WEIGHTED DEPENDENCE GRAPH

In this section, we first define distance between operations of the loop. The distance will indicate the number of iterations that should pass before the value computed by an operation could be used by the dependent operation. For simplicity, it will be assumed that each statement has only one operation. We will denote by $S_r(\overline{i})$, the operation of the statment S_r for the instance $\overline{i}$ of the index vector.

Let $\overline{i} = (i_i,\dots,i_d)$, and $\overline{j} = (j_1,\dots,j_d)$ be two instances of the index vector $\overline{I}$, such that the value computed by $S_r(\overline{i})$ is used as an operand by

$S_t(\bar{j})$. We say that $S_t(\bar{j})$ is data dependent on $S_r(\bar{i})$. We define the distance of $S_t(\bar{j})$ from $S_r(\bar{i})$ (or the length of the dependence arc from S_r to S_t) to be a vector $\bar{m} = \bar{j}-\bar{i} = (j_1 - i_1,\ldots,j_d-i_d)$. The trouble with this definition is that there are $(u_1+1)\ (u_2+1)\ldots(u_d+1)$ N operations in the loop. Certainly, we can not consider an arbitrarily large number of distance vectors. One solution, which is useful also, is te restrict ourselves to those loops for which distance vectors are independent of instances $\bar{i}$ and $\bar{j}$. A representation of a distance vector then could be of form $(m_1,\ldots,m_d)$, where m_i's are either integers, *, or pairs of integers. For example 1, $S_1(\bar{j})$ depends upon $S_2(\bar{i})$ and $\bar{j} = \bar{i} + (2,1)$. In order to assure that distances vectors are independent of indices and their domains, we should assume that for any array variable, the order of appearance of subscripted variables is same for every occurrance of the array variable in the loop, i.e., if A(I-1,J) is an occurrance of an array A, then A(J,I+1) is not allowed, whereas A(I+2,J-1) is. The condition may seem restrictive but most practical examples of loop applications do satisfy it. All examples but one presented in [Bane 79, Padu 80, Kuck 81] satisfy this condition. (is used in distance vectors to represent the case when the corresponding subscripted variable is absent in every occurrance of array variables. If the coefficient of a subscripted variable is not 0, 1 or -1, then we need to store the coefficient also as part of element m_i of the distance vector.

A weighted dependence graph of the loop can be constructed by assigning the distance as the weight of corresponding dependence arcs. The dependence graph of the loop of examples 1 is shown below:



Using the weighted dependence graph, we can trace the complete execution flow graph which represent relationships between operations which compute values and operations that use them as operands. The execution flow graph of example 1 is constructed from its dependence graph. The weighted dependence graphs can be analysed to obtain several run-time properties of the loop, e.g., connecting of execution flow graph and separation of dependent operations, sequencing of operations and their synchronizations.

Our first aim is to develope some condition to examine whether or not the execution flow graph of the loop is weakly connected. The vertex set of the flow graph is the set $\{(i_1,..,i_d):0\leq i_1\leq u_1,...,0\leq i_d\leq u_d\}$ which also represent the set of operations of the loop. In the following theorem, we investigate conditions for partitioning this set into a set of parallel 'planes' such that there are no two operations belonging to different 'planes' which are dependent to one another. In example 1, each such 'plane' is a path imanating from vertices belonging to x or y - axes. For a point $L = (l_1,...,l_d)$ in the domain of the index vector, let I_L, called initial set, denote $\{(i_1,...,i_d):0\leq i_j<l_j$, all $j\pm k$, $1\leq k\leq d\}$.

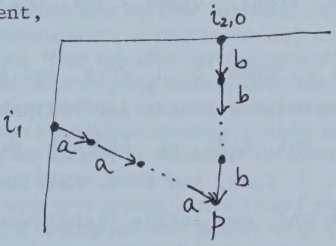
Theorem: A necessary and sufficient condition that operations of a loop can be partitioned into disjoint sets so that (i) operations dependent on an operation belong to the same set, and (ii) for every set of the partition, there is unique operation $S_1(\overline{i})$ that belongs to the set, for some $\overline{i}$ in the initial set I_L for some L, is that the sum of distance vectors of edges of a path that connect a pair of vertices in the dependence graph is the same for every path that connect the pair.

Proof: Sufficiency of the condition is shown first by constructing a partition. A set in the partition can be constructed by traversing edges of the weighted dependence graph so that a vertex is visited only after all its ancestors have been visited, i.e., after including an operation $S_r(\overline{i})$ in a set, $S_t(\overline{i+m})$ is included, where $\overline{m}$ is the distance vector of the arc from S_r to S_t, if all other edges to S_t have been traversed. The dependence graph is traversed continuously over and again (we are, obviously, assuming that the graph contains some cycles) until the value of the index vector runs out of the domain.

Let L be the sum of distance vectors of edges on a path from S_1 to S_1. Since this sum is invariant of paths, this can be computed uniquely. For every unchosen instance $\overline{i}$ in the set I_L, first include the operation $S_1(\overline{i})$ in a set, and then include operations as described above. There may be some operations which may be feeding to $S_1(\overline{i})$. These are operations $S_t(\overline{j})$ such that j is in I_L and $t \neq 1$. These operations can be picked up by traversing the dependence graph once backward. In example 1, the operation C(2,2) feeding to A(4,3) is an example of this type.

We now show that conditions of the theorem are fine. On the contrary, an operation $S_r(\overline{i})$ belong to two sets. Then there must be two operations $S_t(\overline{j})$ and $S_u(\overline{k})$ feeding to $S_r(\overline{i})$, $t \neq u$. Let S_v be the immediate most common ancestor of S_t and S_u in the dependence graph. Because, all paths from S_v to S_r have same length, say $\bar{p}$, then $S_v(\bar{i}-\bar{p})$ must belong to the same sets. Arguing repeatedly, there exists an operation $S_{ri}(\bar{i}_1)$, $\bar{i}_1$ in the initial set, so that it belongs to the same two sets. Because of the construction procedure this is not possible. Hence, the sufficiency of the condition follows.

Now, let the condition be not true, i.e., there are paths of two different lengths between a pair of vertices of the dependence graph. Without loss of generality, there should exist two paths of length, say, a and b from S_1 to S_1. We show that there exists an operation which is reachable from two different instances of $\bar{i}$ satisfying the (ii). To avoid unnecessary notations, we assume that the loop has only two index variables, i.e., the domain of the index vector is a subset of the positive quadrant of x,y-plane. Higher dimensional case is analogues. Following diagram, describe the argument,



i.e., given an instance i_1, there exist an instance $\bar{p}$, such that $S_1(\bar{p})$ is dependent (distantly) on $S_1(\bar{i}_1)$ and $S_1(\bar{i}_2)$, for same another instance i_2. Without loss of generality, let $i_1 = 0$, and let a and b are relative prime. We want to show that there exist integers K_1, K_2, and i_2, such that $K_1a = K_2b + (i_2,0)$, i.e., $S_1(K_1a)$ is dependent on operations $S_1(\bar{0})$ and $S_1((i_2,0))$. Let $a = (x_1,y_1)$ and $b = (x_2,y_2)$. On substituting, $K_1 x_1 - K_2 x_2 = i_2$, $K_1y_1 - K_2y_2 = 0$, which imply that $K_2(\frac{x_1y_2}{y_1} - x_2) = i_2$. If $x_1y_2 - y_1x_2 \neq 0$, then i_2 can be chosen to be $x_1y_2 - y_1x_2$. K_1 and K_2 can be calculated accordingly. If $x_1y_2 - y_1x_2$, then a and b are not prime to each other. The arugment is clearly extendable for loops with any number of index variables. This concludes the proof of the necessity part of the theorem.

In the proof, we have assumed some form of linearity in the use of distance vectors. Linearity holds only if the coefficients of subscripted variables are 1 or -1. Here is a counter example.

Example 2:

```
      DO 2 I = 0, N
      DO 2 J = 0, N
S1:   A(I+1, 2*J+1) = A(I,2J-1) + A(I,2J)
S2:   A(I+1, 2*J+2) = I*J
2     CONTINUE
```

The condition (ii) is not true, though a partition of the execution flow graph exists as before.

REFERENCES

[Kuck 74] D. J. Kuck,P. Budnik, S. C. Chen, E. Davis, Jr., J. R. Han, P. Kraska, D. Lawrice, Y. Muraoka, R. Strebendt, and R. Towle, "Measurement of parallelism in ordinary FORTRAN programs," IEEE Trans. Conput.. Vol. C-23, pp. 37-46, Jan. 1974.

[Bane 79] U. Banerjec, S. C. Chen, D. J. Kuck, and R. A. Towle, "Time and Parallel Processors Bounds for FORTRAN-like loops," IEEE Trans. Comput., Vol. C-28, pp. 660-669, Sept. 1979.

[Padu 80] D. A. Padua, D. J. Kuck, and D. H. Lawrice, "High-Speed Multiprocessors and Compilation Techniques," IEEE Trans. Comput., Vol. C-29, pp. 763-776, Sept. 1980.

[Kuck 81] D. J. Kuck, R. H. Kuhn, D. A. Padua, B. Leasure, and M. Wolfe, "Dependence Graphs and Compiler Optimizations," in Proc. 8th ACM Symp. Principles Prog. Lang., pp. 207-218, Jan. 1981.

[Heuf 82] R. W. Heuft, and W. D. Little, "Improved Time and Parallel Processor Bounds for FORTRAN-like loops," IEEE Trans. Comput., Vol. C-31, pp. 78-81, Jan. 1982.

VLSI IMPLEMENTATIONS OF GEOMETRIC TASKS.

BERNARD CHAZELLE
Computer Science Department
Brown University
Providence, RI 02912

Abstract

This paper describes systolic algorithms for a number of geometric problems. Implementations yielding maximal throughput are given for solving dynamic versions of convex hull, inclusion, range and inverse range search, planar point location, intersection, triangulation, and closest-point problems.

1. Introduction

The pervasive influence of VLSI in the computer science community has given research on parallel computation its second wind. In contrast with the traditional conception of parallel systems, where several computers are each assigned complicated tasks, VLSI computation, especially of systolic nature, involves the simultaneous use of a great number of very simple processors [MC,K]. As commonly referred to, systolic arrays are one- or two-dimensional arrangements of simple cells locally connected [K,K1,KL,L]). The essential features of systolic cells are their *simplicity*, *regularity*, and *modularity*. Performance-wise, these characteristics are definite assets, as they ensure high levels of *pipelining* and *multiprocessing*, hence providing massive parallelism. They also affect the economics of the approach by making circuit development more cost-effective. Indeed, with dropping costs of electronic components and increasing levels of circuit integration, systems designers are facing the prospect of putting hundreds of thousands of gates on a single chip, which so far constitutes a formidable challenge. Systolic architectures are one answer to this challenge. Their modularity permits the designer to decompose the system's architecture into building blocks which can be used repetitively with simple interfaces.

The purpose of this work is to present a *class-related* systolic processor based on the approach just described. This processor is a programmable systolic array aimed for solving a wide class of geometric problems in a highly unifying manner. This *class* of problems contains many of the most basic questions of computational geometry. Among others, we will find dynamic versions of convex hull, inclusion, range and inverse range search, planar point location, intersection, triangulation, and closest-point problems. Whenever possible, we will insist on the dynamic aspect of the problem, for it is often where systolic solutions are at their best. On the other hand, many applications areas involve problems of an inherently dynamic nature, with which we must cope. For example, air traffic control necessitates the real-time solution of closest-point problems on an ever-changing set of points.

After discussing the advantages of systolic architectures in terms of increased adaptability and cost-effectiveness, we should investigate the gains in performance to expect from a systolic treatment of computational geometry. To begin with, let us roughly describe our systolic architecture. We consider only one-dimensional arrays, i.e., arrays with a single string of cells, each connected to their one or two neighbors. Furthermore, communications with the outside world (typically, a host computer) takes place solely at either of the end-cells. It results from this configuration that although there may be full parallelism in the arrays, the number of I/O operations at any time is always bounded by a constant. We do not make this assumption for the sake of simplicity, but for the sake of *realism*, since in most applications, the systolic device will receive its data from a sequential computer. It is often the case that a circuit will receive streams of data, each of them pertaining to a different instance of the problem. In this case, maximizing the throughput is called *pipelining*, and to measure the adequacy of the circuit to respond to a stream of requests, we look at its *period*, a concept introduced in [VU]. Roughly, the period of a circuit is the minimum delay between two consecutive sets of inputs. Of course, it is highly desirable that our systolic designs have period $O(1)$, which often involves preventing the occurrence of clusters or the presence of cells waiting for others in order to complete execution.

In the next section, we describe the general features of the geometric systolic array, then proceed with a detailed description of the algorithms in the remaining sections. Because of the intricacy of some of these algorithms, we had to keep the descriptions at a fairly intuitive level. The detailed algorithms, however, can be found in a companion paper [CH].

2. The Geometric Systolic Chip

Most of the systolic arrays which we will describe in this paper have the basic outlook of fig.1. Interaction with the outside world takes place solely at the end cells, called *boundary* cells. All of the other cells, called *generic*, are alike, and although boundary cells are assigned additional tasks for I/O purposes, they usually don't differ drastically from the generic cells. Each cell contains a small amount of memory, in the form of a few registers. We distinguish two kinds of registers:

1. Working registers for either storing data (point, edge, angle,...) or for providing temporary storage for the computations.
2. I/O registers for communicating data between adjacent cells.

To avoid dealing with implementation details, we may regard I/O registers as being conceptually "located" on the connection wires between the cells. These registers are protected by gates which can be either *open* or *locked* according to the current clock phase. We assume that the whole systolic array is synchronous, and that each cell operates in lock-step. For simplicity, we also assume the existence of two clocks ϕ_1 and ϕ_2 beating in opposition. This allows us to separate input and output stages easily by requiring that input (resp. output) gates should all be open (resp. locked) at ϕ_1 and vice versa at ϕ_2 (fig.2). The lapse of time between two phases ϕ_1 is called a *systolic cycle*. It is to be distinguished from the clock cycle internal to each cell, which is likely to be much shorter. Indeed, a systolic cycle must correspond at least to a number of internal clock cycles necessary for a cell to complete the execution of its stored program. We should observe that this clocking arrangement is not unique; systolic arrays with asynchronous and/or adjacent cells operating in opposite cycles are perfectly feasible, so the choice made here serves only explanatory purposes, wlog. The only bit of notation, used throughout, that needs be introduced here concerns the representation of points by capital letters, A,M,X,..., with a_i denoting the ith coordinate of point A in a Cartesian system of coordinates.

3. Convex Hull Problems

Estimating a population parameter in statistics, or simulating chemical reactions often require computing the convex hull of a set of points in a dynamic fashion [S]. For static and dynamic solutions to convex hull problems on a conventional machine, see [S,P1,LE,J,OV]. To fulfill our purposes, we will devise a systolic structure which supports the following operations.

1. Insert/delete point M.
2. Find and report all the vertices of the convex hull in clockwise or counterclockwise order.
3. Determine whether an arbitrary point M lies inside or outside the convex hull.

As usual with dynamic convex hull routines, deletions and insertions proceed in very different ways. To cope with this problem, we will describe two systolic arrays, **CH1** and **CH2**, supporting the following operations.

Array CH1

1. Insert/delete point M.
2. Report all vertices of convex hull (in arbitrary order).
3. Determine whether point M lies inside or outside the convex hull.

Array CH2

1. Insert point M.
2. Report all vertices of convex hull in clockwise (or counterclockwise) order.
3. Determine whether point M lies inside or outside the convex hull.

We observe that in order to support the operations listed at the beginning, it suffices to connect **CH1** and **CH2** together.

3.1 The array CH1

CH1 consists of N cells, so as to handle up to N points at any given time, each cell storing one point. All operations (updates and queries) are initiated at the input cell with the answers emanating from the output cell (fig.3).

Implementing Operation 1 is straightforward. Points to be inserted are pumped into the left cell, and travel from left to right stopping at the first vacant cell. A point to be deleted is input in the same way, moving from left to right until it encounters the cell where its copy is stored, which it then marks as vacant. Note that the array does not keep track of the order of the vertices around the convex hull. Operation 3 relies on the following geometric property.

Lemma 1: Let $M_0,\ldots,M_{N-1}$ be a list of N points in the order induced by an angular sweep around a point M. This point lies inside the convex hull of $M_0,\ldots,M_{N-1}$ if and only if no angle of the form (M_iMM_{i+1}) (mod N) exceeds 180 degrees.

Proof: A consequence of the fact that a point lies outside the convex hull iff there exists a line containing it, with all the points on one side of the convex hull. ∎

Lemma 1 shows that we simply have to make the query point M travel from left to right, maintaining the value of the largest angle (M_iMM_{i+1}) encountered so far. This is done by a trivial case analysis, illustrated in fig.4. To alleviate the notation, we define F(M,A,B) as the sign of the expression $um_1 + vm_2 + w$, where $uX + vY + w = 0$ is an equation of the line passing through A and B. This provides us with an easy characterization of whether two points M,P lie on the same side of AB, i.e., they do iff F(M,A,B)=F(P,A,B). For simplicity, we will always assume that no three points are ever collinear. Let T=(M,A,B) be the triplet of points yielding the largest angle so far. M will travel along with this piece of information, which must be tested against each new point encountered, then updated before proceeding to another cell. Testing T against a new point C leads to the operations described in fig.4.

The handling of Operation 3 should be clear by now, so can proceed with Operation 2. One solution would be, in a first stage, to output copies of all the points, then in a second stage, re-input them one after the other, while executing Operation 3. To achieve the same result *in place*, we can view the systolic array as a strip of paper. The idea is then to pick it up at the input cell end and fold it over, pulling the input cell over from left to right (fig.5). To ensure that each cell will indeed look at all the others, we must update both the covering cells moving right and the covered cells not yet in motion. The updating is of the same nature as in Operation 3. Note that the left end of the folded strip will move twice as slowly as the input cell. For this reason, no operation on the systolic array should be initiated within N systolic cycles after the start of Operation 2. This will ensure that no query will ever propagate to a cell already engaged in a computation for a previous query. To implement this *fold-over* operation, we need essentially two signals: one is the query itself, which follows the right-end of the covering strip. The other follows the other end, and is necessary to signal the cell that at the cycle following the next, it will have to send a copy of itself to the right, thus becoming the current left front of the covering strip.

3.2 The array CH2

This structure supports only insertions, but in return, it provides an ordered description of the convex hull, at any time. Also, since the array stores only the vertices of the convex hull, it can support an arbitrary number of insertions, as long as this convex hull always keeps a number of vertices on the order of N. To begin with, let us give the geometric background behind the algorithm. Assume that $M_0,\ldots,M_{p-1}$ are the vertices of a convex p-gon P, given in clockwise order. Let M be an arbitrary point outside P, and let Q denote the convex hull of $P \bigcup \{M\}$. Considering the infinite line passing through an edge e of P, it is easy to see that adding M to the convex hull will cause the disappearance of e if and only if the line lies between M and P. This motivates the introduction of the function G, defined by the relation:

$$G(M,A,B) = (a_2 - b_2)m_1 + (b_1 - a_1)m_2 + a_1b_2 - a_2b_1.$$

Note that $F(M,A,B) =$ sign $G(M,A,B)$. The following result is simply a more formal statement of the remark above, and we leave out the proof — see illustration in fig.6. Once again, in the following, we shall assume that no three points may be collinear.

Lemma 2: Let $M_0,\ldots,M_{p-1}$ be the vertices of a convex p-gon P, in clockwise order. Let M be an arbitrary point and Q denote the convex hull of $P \bigcup \{M\}$.

1. M lies inside P iff $G(M, M_i, M_{i+1}) < 0$, for all i; $0 \leq i \leq p-1 \pmod{p}$.
2. M_iM_{i+1} is an edge of Q iff $G(M,M_i,M_{i+1}) < 0$. Also, if M does not lie inside P, it is a vertex of Q and its adjacent vertices are, in clockwise order, M_u and M_v, defined uniquely by $G(M_{u-1},M_u,M) < 0$, $G(M_{u+1},M_u,M) < 0$, $G(M_{v-1},M,M_v) < 0$, and $G(M_{v+1},M,M_v) < 0$.

The array **CH2** has the same overall structure as **CH1** (fig.3). Instead of a point, each cell now stores an edge of the convex hull, however, and the left-to-right order in the array corresponds to a clockwise traversal of the boundary of the convex hull. Operation 1 (*inserting point M*) causes M to travel from the input cell to the output cell, computing the function G defined above in order to determine whether M lies inside the convex hull. If it lies outside, two edges have to be added to the structure, and in general, a bunch of consecutive edges (at least one, anyhow) must be removed. More precisely, assume that $M_iM_{i+1},\ldots,M_{j-1}M_j$ are the consecutive edges of P to be removed. Upon encountering M_iM_{i+1}, M must cause the cell currently visited to substitute M_iM for M_iM_{i+1}. All the subsequent cells will delete their contents, until M encounters the first edge (M_jM_{j+1}) not to be affected by the insertion of M. At this point, the current cell must hand the cell M_jM_{j+1} to its right-hand side neighbor, and keep the edge MM_j in store. M has now ceased to cause changes in the array, and it can terminate its motion. However, there is now one cell in the array with two edges. To repair this anomaly, we make sure that the cell keeps its additional edge but forward its former contents to its right neighbor. This only causes to shift the anomaly one cell to the right, but iterating on this process will eventually cause the last non-vacant cell to release an edge to its neighbor, which solves the problem. This phenomenon is known as ***rippling***, as it mimics the propagation of a wave in water. We should observe that if the last non-vacant cell has no right neighbor, ***overflow*** must be reported. However, the insertion may have just cause the deletion of a number of edges, in which case reporting overflow is undesirable. In general, we pose as a requirement that ***(no overflow should be reported if there is any vacant cell in the array)***, no matter where. To comply with this rule, we must ensure that vacant cells which have edges on their right-hand side, i.e., ***holes***, must be ***filled*** by edges from the right. To do so, it suffices to have each cell always check whether its left hand-side neighbor is vacant, in which case it must pass its contents to it. As a result, it appears that, in general, two opposite motions will take place within the array: one, to the right, corresponds to queries and insertions, while the other, leftwards, is meant to fill the holes just created. Operation 2 simply involves pumping out all the edges of the array through the input cell, thus preserving the (counterclockwise) order of the edges. Operation 3 is a simple application of Lemma 2, similar to Operation 1, yet without altering the state of the array. The query point M travels left-to-right, checking its location with respect to each edge in turn. If M is always found to lie on the same side of the edge as the interior of the polygon, inclusion must be reported, otherwise M lies outside the convex hull.

4. Inclusion, Intersection, and Closest-point problems

We next show that many of the most common geometric problems can be solved by means of a simple unifying scheme. The underlying idea, already used in arithmetic or pattern matching [K,FK,KL], exploits the inherent suitability of systolic designs to testing each input data against the contents of each cell, in a pipeline fashion. More precisely, let $S_1,\dots,S_N$ be the data stored in the array, and let $a_1,\dots,a_j$ denote a list of queries in the order with which they arrive at the input cell: the goal is to compute for each query a_k the value of $T_k^{(N)}$, defined by the recurrence relation: $T_k^{(1)} = 0$,

$$T_k^{(i)} = F(T_k^{(i-1)}, S_i, a_k).$$

Figure 7 sketches a systolic solution for this class of problems. As we will see, it is possible, in most cases, to make the systolic scheme dynamic, that is, capable of handling updates in the array. If no order among the S_i's is required, a ***delete*** (e) operation simply results in marking the cell storing e vacant, while ***insert*** (e) causes the storing of e in the first cell vacant from the left. If on the other hand, some order is to be preserved among the S_i's, an ***insert*** operation will involve searching for the appropriate (non-necessarily vacant) cell, and store the new element in it, thus possibly causing the remaining cells to ripple to the right. Symmetrically, deleting an element will incur the creation of a hole and the start of a leftward motion aimed at filling it, resulting in the propagation of the hole to the right end of the array.

4.1 Inclusion Problems

1) Point / Polygon

Does point M lie in polygon $P = M_1,\dots,M_N$?

The polygon is taken to be simple, but no convexity assumptions are made. It is possible to achieve unit period with the following systolic scheme. The register S_i holds the pair (M_i,M_{i+1}), where the list $M_1,\dots,M_N$ corresponds to a clockwise traversal of the boundary of P. The variables **x** and **y** of fig.7 are respectively the point M and the pair (uv,u'v'), where uv and u'v' are the edges of P with u,v (resp. u',v') giving the clockwise direction, such that their intersection with the vertical line L passing through M forms the smallest segment so far containing M (fig.8). Testing for the inclusion of point M involves pumping M throughout the array, from left to right, updating the pair of edges in **y** on the fly.

```
Let $y_{in}$ = (AA',BB'), with $D_A = L \cap AA'$ and $D_B = L \cap BB'$.
Let $D = M_iM_{i+1} \cap L$
if D lies on $D_AD_B$
   then
      if M lies on $D_AD$
         then $y_{out} \leftarrow (AA',M_iM_{i+1})$
         else $y_{out} \leftarrow (BB',M_iM_{i+1})$ .
   else
      $y_{out} \leftarrow y_{in}$
$x_{out} \leftarrow x_{in}$
```

If D_A (resp. D_B) is undefined, it can be set to $+\infty$(resp. $-\infty$), for convenience.

Eventually, the array can output an inclusion message if y_{out} falls into case b) of fig.8, or a non-inclusion signal if it falls into case c). Note that the scheme used above is far from unique, and other tests for inclusion may lead to equally simple systolic structures. For example, simply counting the number of intersections with the line L above and below M is sufficient, since these numbers are even iff M lies inside the polygon.

2) Planar point location

Given a planar graph with faces $f_1, \ldots, f_N$, and a point M, determine the face where M lies.

For this problem, several sequential algorithms with an optimal $O(\log N)$ query time exist [S,LT,P2], but for the most part, require complicated preprocessing. Instead, we can design a very simple systolic array to solve this problem with unit period. To do so, we simply represent the graph by placing in the array, next to each other, clockwise descriptions of the faces. Since in this way, each edge is represented exactly twice, and since the total number of edges of a planar graph does not exceed the number of faces, up to within a constant factor, no more than a linear number of cells will be required. We can now view the graph as a union of polygons, represented in the array by consecutive sublists of edges. Locating a query point M comes down to testing the point for inclusion with respect to each polygon in turn, as previously described, finally concluding with a report of the name of the unique polygon which contains M.

3) Range search

In one dimension, the problem consists of computing the number of segments containing a query point, given a set of N collinear segments. In two dimensions, the goal is to report the number of rectangles containing a query point, given a set of N iso-rectangles (sides parallel to the X-Y-axes) [BW,BO,NP,E,M].

The systolic array will simply store one segment (resp. rectangle) in each cell, so that the query point can scan the array left-to-right, checking for inclusion in the segment (resp. rectangle) stored in each cell, and updating the partial count. Note that the problem can be extended to arbitrary polygons instead of only iso-rectangles.

4) Inverse range search

Given a set of segments (resp. rectangles), and given a query segment (or a query rectangle), report the number of segments (resp. rectangles) that intersect the query object [BW,BO,NP,E,M].

Once again, testing pairwise intersection requires constant time, which ensures unit period. The algorithm is straightforward and needs no further development. The last two problems arise constantly in graphics [NS], and in design-rule checking for VLSI circuits [BO,BW]. Often, however, instead of a mere number of intersections, an explicit report of all the intersecting pairs is desired. To give our systolic arrays this added capability, it is sufficient to add only a few instructions to the algorithms. One solution is to prescribe that upon encountering an intersection, a query first sends the intersecting pair forward to the next cell, then only proceeds in the same direction. Of course, this will cause a slowdown, therefore to prevent overtaking by subsequent queries, we require that before moving an object to the next cell, the algorithm first check the vacancy of that cell. To that end, each cell must keep sending *vacant* or *occupied* signals to its left hand-side neighbor. The scheme is somewhat similar to the traffic management described for **CH2**. We should observe that with the actual reporting of intersecting pairs, the array still yields maximal throughput, since the output flow is always kept at its maximum. The concept of period, based on *input* rate, becomes meaningless, however, since a glut due to intense output activity may cause a slowdown in the input rate.

4.2 Intersection Problems

For sequential algorithms, see [S,SH,BW,BO,NP].

5) Intersection of polygons

Given two polygons P, Q, determine whether they intersect.

If we wish to determine only if the boundaries intersect, we may simply store the edges of P in the systolic array, and have those of Q travel left-to-right, testing each edge encountered for intersection. It is easy to extend the method and solve the general problem by observing that P and Q intersect if and only if at least one of the following conditions is satisfied:

1. A vertex of P lies in Q.
2. A vertex of Q lies in P.
3. The boundaries of P and Q intersect.

Thus it suffices to add to each cell two copies of the procedure described for Problem 1); one with respect to P, the other with respect to Q. Note that each cell must check whether the passing edge is the last edge of Q, in which case, it must tag a *yes* or *no* signal to the tail of Q to acknowledge if either endpoint of the edge stored in the cell lies inside Q or not. This is straightforward, and details are left to the attention of the reader.

6) Intersection of half-planes

Given N half-planes $H_1, \ldots, H_N$, compute their intersection.

This problem requires $\Omega(N \log N)$ time on a conventional machine [S,SH,B]. As usual, we expect our systolic implementation to yield maximal throughput, and thus display an overall $O(N)$ time performance. Moreover, as we will see, it is easy to provide the array with the capability of handling queries and updates, without losing on the overall performance. This addition is very similar to the connection of **CH1** and **CH2** described earlier for the solution of dynamic convex hull problems. Actually, the similarity between the two problems is very deep, for it stems from the geometric duality which exists between convex hulls and intersections of half-spaces [B,PM]. Let I be the intersection of the N half-planes $H_1, \ldots, H_N$. If I is not empty, it is a convex polygon with possibly one open side, i.e., two edges that are half-lines meeting at infinity. It is possible to represent I either by a list of the lines supporting the edges of I, in arbitrary order, or if we wish more information, by a list L of edges (A,B), as they appear in a clockwise traversal of the boundary. In case of an open polygon I, we require that the vertex at infinity should appear at the ends of the list. For example, we may have two points I_1, I_k, in the list

$$L = \{(I_1, A_1), (A_1, A_2), \ldots, (A_k, I_k)\}$$

with the understanding that the edge I_1A_1 (resp. A_kI_k) is the infinite ray starting at A_1 (resp. A_k) and passing through I_1A_1 (resp. A_kI_k). Similarly to **CH1** and **CH2**, we will design two systolic arrays **INT1** and **INT2** to support the following operations:

Array **INT1**

1. Insert/delete half-plane H.
2. Report all lines on the boundary of I, in arbitrary order.
3. Determine whether point M lies in I.

Array **INT2**

1. Insert half-plane H.
2. Report all vertices of I in clockwise (or counterclockwise) order.
3. Determine whether point M lies in I.

Because of the similarity with **CH1** and **CH2**, we may only sketch the algorithms. Any standard representation of half-planes is adequate. For example, $(u, v, w, \geq)$ can be used to denote the half-plane $uX + vY + w \geq 0$. The only point to investigate about **INT1** is, in Operation 2, the type of matching involved in the "fold-over" process. To begin with, it is easy to see that a half-plane H_i contributes an edge to I iff its supporting line L_i lies in the intersection of the $N - 1$ remaining half-planes $H_1, \ldots, H_{i-1}, H_{i+1}, \ldots, H_N$. Then since the intersection of L_i with the intersection of any subset of $H_1, \ldots, H_N$ is, if not empty, a segment, a half-line, or L_i itself, it can be expressed by means of at most two points, which can then be updated as L_i is matched against each H_j in turn. All of the other features of **INT1** are similar to those of **CH1**. As for **INT2**, we assume that, at all times, the array contains a clockwise description of I, with each edge stored in a separate cell. Once again, all the operations are handled as in **CH2**, including the hole-filling process.

4.3 Closest-point Problems

7) Nearest-neighbor

Given N points, $M_1, \ldots, M_N$, and a query point M, determine the nearest neighbor of M — see [S,BSW, BWY].

For this problem, we allow the dimension of the space to be arbitrary and the distance to be based on any of the $L_1, L_2, \ldots, L_\infty$. Whereas efficient solutions on a conventional machine involve the use of fancy data structures (e.g., *Voronoi diagrams, planar point location search trees, k-d trees, etc...*) entailing substantial implementation overhead, a simple dynamic systolic scheme can be devised as follows: Once again, we store one point per cell. Queries travel left-to-right, determining their nearest neighbor on the fly. To do so, each query is accompanied by the the closest point found so far. Updates in the structure are handled as in **CH1**, that is, inserting a point into the first available cell encountered, and deleting it by simply marking the corresponding cell vacant. If desired, a *report-all-nearest-neighbors* query can be added to the set of allowed operations. This instruction, which causes the nearest neighbor of each point in the array to be output, can be implemented by the *fold-over* procedure of **CH1**. Applications areas where a device for reporting near-neighbors would be of great interest are many. Air traffic control is one example: in this situation, typically, a few radars transmit streams of signals giving updates on the position of near-by airplanes, and minimum safety distances between planes must be constantly ensured. To speed up the signaling of anomalous positions, an *emergency* output port can be reserved on each cell, with direct link to the host. Although slightly unsystolic, this feature is totally feasible as long as emergency reports remain rare events.

8) Euclidean minimum spanning tree

Given N points in the plane, construct a tree of minimum total length whose vertices are the given points [S].

C. Savage, in [SA], proposes a systolic structure for computing the connected components of a graph. This structure is a one-dimensional systolic array, which can be connected to Leiserson's priority queue [L], so as to compute the minimum spanning tree in linear time.

9) Triangulation

Partition the convex hull of N points $M_1, \ldots, M_N$ into triangles, using only segments between the points.

This problem, which arises frequently in numerical analysis (*finite element method, numerical interpolations, etc...*), has an $\Omega(N \log N)$ lower bound on a sequential machine [S]. A one-dimensional systolic scheme can yet achieve linear time, while supporting the following features.

Array **TRI**

1. Insert a point in the triangulation.
2. Determine in which face of the triangulation a query point lies.
3. Report all the triangles of the triangulation by giving, for each, a clockwise order list of its vertices.

The array **TRI** computes an arbitrary triangulation, without any consideration of "goodness." Since in many cases, however, it is crucial that certain quality criteria are met, e.g, minimizing a function of the edges, the array might be used more advantageously within the framework of a more complicated heuristic. Each occupied cell may serve one of two purposes: either it stores an edge of the convex hull (R=(A,B)) with A,B giving the clockwise orientation, or it stores the vertices of a triangle in clockwise order. We also require that, from left to right, the edges stored in the cells of the first kind should appear in clockwise order (fig.9). Finally we assume the existence of a flag **F** to signal the first edge of either the upper or the lower chain. With this arrangement, Operation 2 simply involves testing the query point against each triangle, carrying the containing triangle along with M, when detected (if ever), otherwise reporting an *outsideface* message, if no such triangle has been found. Yet simpler, Operation 3 involves pumping out the contents of each cell storing a triangle, one by one.

To handle Operation 1, two cases must be considered:

1. M lies inside a triangle (e.g., DFC in fig.9). We must replace R by, say, MCD, and insert the triangles MDF and MFC into the next two right neighbors of the current cell. This is done by rippling to the right (fig.9 – case 1).
2. M lies outside the convex hull, and thus will become a vertex of the new convex hull. The algorithm is very similar to **CH2**. Instead of deleting non-convex-hull edges, however, we must now insert new triangles into the array.

5. Conclusions

The purpose of this work has been to present systolic designs for several geometric problems. Most of the algorithms described in this paper involve two distinct types of tasks. One is concerned with the actual computation of geometric functions, and is, in general, the easier to understand. The other involves initiating and granting requests, which entails moving data around, i.e., adding new items into the array or filling holes created by deletions. In general, the flow of data is irregular and not predetermined, since it is contents-dependent. With the exception of priority queues and similar structures [GL,L], this constitutes a major departure from most systolic arrays described in the literature, especially those for arithmetic computations [KL,K,FK]. Instead, most of the known systolic structures have a fixed, predetermined data flow, usually highly regular. One major difficulty with *random* motion is the absence of adequate tools for proving the correctness of the algorithms, and in particular, describing the behavior of the data flow. There certainly lie promising avenues of research.

Because of space limitations, we must refer the reader to [CH] for details on the algorithms sketched in this paper. At the implementation level, the one observation we will make, however, is to stay away from floating-point representations, whenever possible, because of the inevitable complications which they entail. Note that in all the algorithms mentioned, only fixed-point additions, subtractions, and multiplications are needed, with the exception of the intersection algorithms, which involve the solution of linear equations. In this case, division is needed, yet can be avoided, if rational numbers are kept as pairs of fixed-point numbers, as is common practice in linear programming. We should also observe that the arithmetic computations involved in the algorithms are in general very simple and limited, most of them consisting of simple fixed-point inner products.

Future work in the area of systolic algorithms includes, of course, their actual implèmentation and evaluation. Also, any attempt at classifying the problems that lend themselves to systolic implementations appears very worthwhile. Finally, we must once again emphasize the current need for an original description language for systolic systems, as well as new tools for studying the behavior and proving the correctness of the underlying algorithms.

Acknowledgments

I wish to thank H.T. Kung of CMU for motivating this research, and Janet Incerpi of Brown University for her invaluable assistance in the final preparation of this manuscript.

This research was sponsored by DARPA Order No. 3597, monitored by the Air Force Avionics Laboratory Under Contract F33615-81-K-1539.

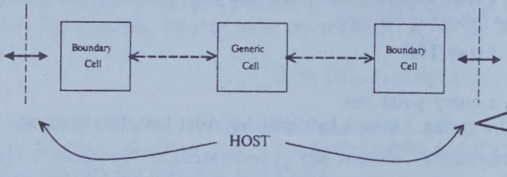

Boundary Cell
Generic Cell
Boundary Cell
HOST

Figure 1: The one-dimensional systolic array.

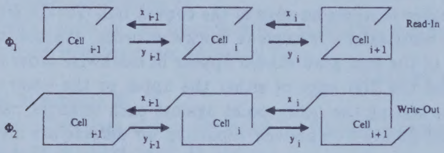

Φ_1
Cell i-1
x_{i-1}
y_{i-1}
Cell i
x_i
y_i
Cell i+1
Read-In
Φ_2
Cell i-1
x_{i-1}
y_{i-1}
Cell i
x_i
y_i
Cell i+1
Write-Out

Figure 2: Handling critical paths.

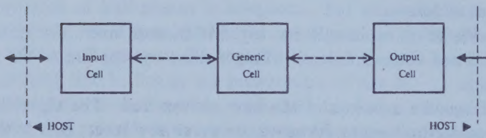

Input Cell
Generic Cell
Output Cell
HOST
HOST

Figure 3: The overall structure of the array CH1.

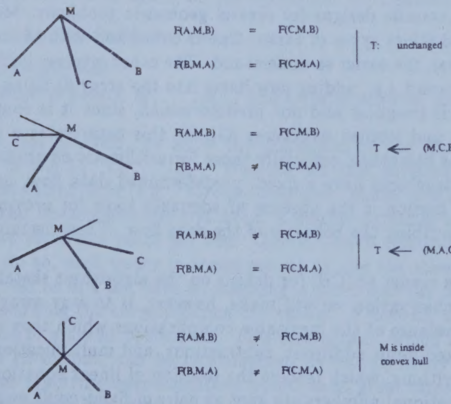

M
A
C
B
F(A,M,B) = F(C,M,B)
F(B,M,A) = F(C,M,A)
T: unchanged
M
C
A
B
F(A,M,B) = F(C,M,B)
F(B,M,A) ≠ F(C,M,A)
T ← (M,C,B)
M
A
C
B
F(A,M,B) ≠ F(C,M,B)
F(B,M,A) = F(C,M,A)
T ← (M,A,C)
C
M
A
B
F(A,M,B) ≠ F(C,M,B)
F(B,M,A) ≠ F(C,M,A)
M is inside convex hull

Figure 4: Testing inclusion in the convex hull.

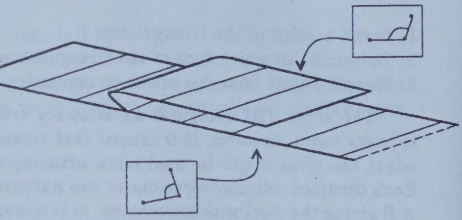


Figure 5: The fold-over operation.

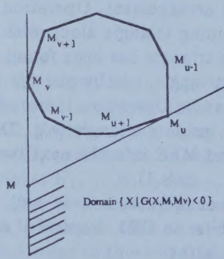

M_{v+1}
M_{u-1}
M_v
M_{v-1}
M_{u+1}
M_u
M
Domain { X | G(X,M,Mv) < 0 }

Figure 6: Computing convex hulls in clockwise order.

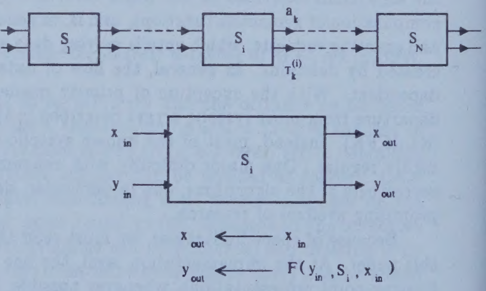

S_1
S_i
S_N
a_k
$T_k^{(i)}$
x_{in}
y_{in}
S_i
x_{out}
y_{out}
$x_{out} \leftarrow x_{in}$
$y_{out} \leftarrow F(y_{in}, S_i, x_{in})$

Figure 7: A systolic scheme for iterative problems.

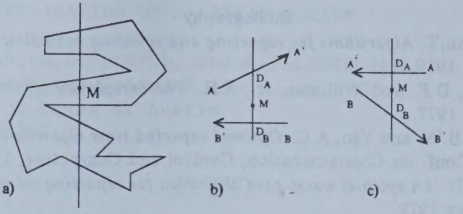

M
a)
A'
A
D_A
M
B'
D_B
B
b)
A'
D_A
A
M
B
D_B
B'
c)

Figure 8: Testing inclusion.

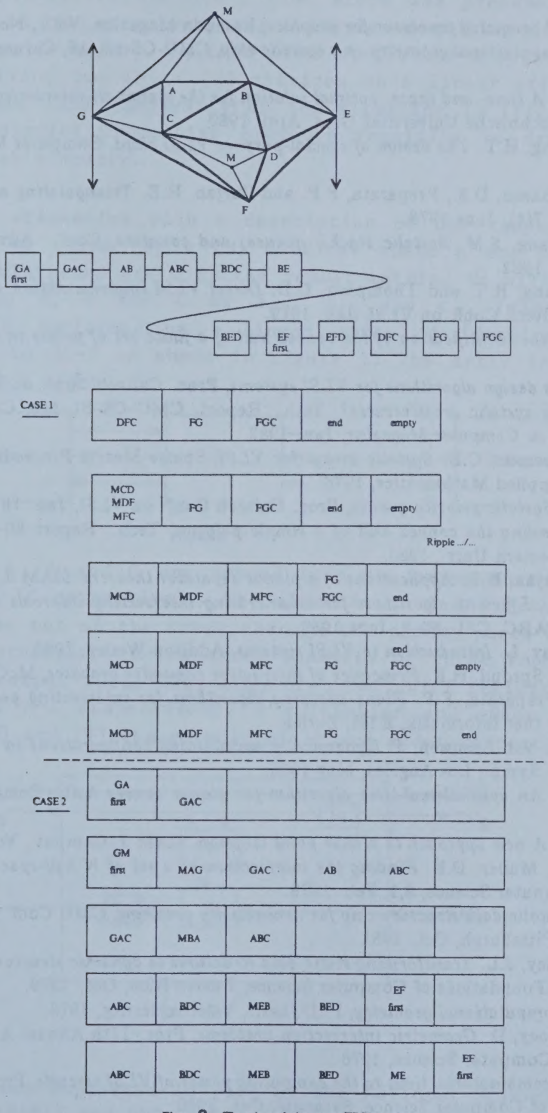

M
A
B
G
C
E
M
D
F
GA first
GAC
AB
ABC
BDC
BE
BED
EF first
EFD
DFC
FG
FGC
CASE 1
DFC
FG
FGC
end
empty
MCD MDF MFC
FG
FGC
end
empty
Ripple .../...
MCD
MDF
MFC
FG FGC
end
MCD
MDF
MFC
FG
end FGC
empty
MCD
MDF
MFC
FG
FGC
end
CASE 2
GA first
GAC
GM first
MAG
GAC
AB
ABC
GAC
MBA
ABC
ABC
BDC
MEB
BED
EF first
ABC
BDC
MEB
BED
ME
EF first

Figure 9: The triangulation array TRI in action.

Bibliography

[BO] Bentley, J.L., Ottmann,T. *Algorithms for reporting and counting geometric intersections*, IEEE Trans. Comp. C–28,9, Sept. 1979.

[BSW] Bentley, J.L., Stanat, D.F. and Williams, Jr., E.H. *The complexity of near-neighbor searching*, Info. Proc. Lett. 6,6, Dec. 1977.

[BWY] Bentley, J.L., Weide, B.W. and Yao, A.C. *Optimal expected-time algorithms for closest-point problems*, Proc. 16th Allerton Conf. on Communication, Control and Computing, 1978.

[BW] Bentley, J.L., Wood, D. *An optimal worst-case algorithm for reporting intersections of rectangles*, Rep. CMU–CS–79–122, May 1979.

[B] Brown, K.Q. *Geometric transforms for fast geometric algorithms*, PhD thesis, Carnegie-Mellon Univ., 1979.

[C] Clark, J. *A VLSI geometry processor for graphics*, Lambda Magazine, Vol.1, No.2, 1980.

[CH] Chazelle, B., *Computational geometry on a systolic chip*, CMU-CS-82-119, Carnegie-Mellon University, 1982.

[E] Edelsbrunner, H. *A time- and space- optimal solution for the planar all-intersecting-rectangles problem*, Tech. Report., Technische Universitat Graz, April 1980.

[FK] Foster, M.J., Kung, H.T. *The design of special-purpose VLSI chips*, Computer Magazine, 13 (1), Jan. 1980.

[GJ] Garey, M.R., Johnson, D.S., Preparata, F.P. and Tarjan, R.E. *Triangulating a simple polygon*, Info. Proc. Lett., Vol. 7(4), June 1978.

[GL] Guibas, L.J., Liang, S.M. *Systolic stacks, queues, and counters*, Conf. Adv. Res. VLSI, MIT, Cambridge, Jan. 1982.

[GKT] Guibas, L.J., Kung, H.T. and Thompson, C.D. *Direct VLSI implementation of combinatorial algorithms*, Proc. Caltech Conf. on VLSI, Jan. 1979.

[J] Jarvis, R.A. *On the identification of the convex hull of a finite set of points in the plane*, Info. Proc. Lett. 2, 1973.

[K] Kung, H.T. *Let's design algorithms for VLSI systems*, Proc. Caltech Conf. on VLSI, Jan. 1979.

[K1] Kung, H.T. *Why systolic architectures?*, Tech. Report, CMU–CS–81–148, Carnegie-Mellon Univ., Nov. 1981. Also in *Computer Magazine*, Jan. 1982.

[KL] Kung, H.T., Leiserson, C.E. *Systolic arrays for VLSI*, Sparse Matrix Proceedings 1978, Society for Industrial and Applied Mathematics, 1978.

[L] Leiserson, C.E. *Systolic priority queues*, Proc. Caltech Conf. on VLSI, Jan. 1979.

[LE] Lee, D.T. *On finding the convex hull of a simple polygon*, Tech. Report 80–03–FC–01, Dept. of EE&CS, Northwestern Univ., 1980.

[LT] Lipton, R.J., Tarjan, R.E. *Applications of a planar separator theorem*, SIAM J. Comp., 9 (3), 1980.

[M] McCreight, E.M. *Efficient algorithms for enumerating intersecting intervals and rectangles*, Tech. Report. Xerox PARC, CSL–80–9, June 1980.

[MC] Mead, C., Conway, L. *Introduction to VLSI systems*, Addison-Wesley, 1980.

[NS] Newman, W.M., Sproull, R.F. *Principles of interactive computer graphics*, McGraw-Hill, 1973.

[NP] Nievergelt, J., Preparata, F.P. *Plane-sweeping algorithms for intersecting geometric figures*, Tech. Report., Institut fuer Informatik, ETH, Zurich.

[OV] Overmars, M.H., Van Leeuwen, J. *Dynamically maintaining configurations in the plane*, Proc. 12th Annual SIGACT Symp., Los Angeles, May 1980.

[P1] Preparata, F.P. *An optimal real-time algorithm for planar convex hulls*, Comm. ACM, 22, 7, July 1979.

[P2] Preparata, F.P. *A new approach to planar point location*, SIAM J. Comput., Vol. 10 (3), Aug. 1981.

[PM] Preparata, F.P., Muller, D.E. *Finding the intersection of a set of N half-spaces in time O(Nlog N)*, Theoretical Computer Science, 8,1, Feb. 1979.

[SA] Savage, C. *A systolic data structure chip for connectivity problems*, CMU Conf. on VLSI Systems and Computations, Pittsburgh, Oct. 1981.

[SB] Saxe, J.B., Bentley, J.L. *Transforming static data structures to dynamic structures*, Proc. 20th Annual IEEE Symp. on Foundations of Computer Science, Puerto Rico, Oct. 1979.

[S] Shamos, M.I. *Computational geometry*, PhD thesis, Yale University, 1978.

[SH] Shamos, M.I., Hoey, D. *Geometric intersection problems*, Proc. 17th Annual Annual IEEE Symp. on Foundations of Computer Science, 1976.

[VU] Vuillemin, J. *A combinatorial limit to the computing power of VLSI circuits*, Proc. 21st Annual Symp. on Foundations of Computer Science, Syracuse, Oct. 1980.

SYSTOLIC MATRIX MULTIPLICATION ON A LINEAR ARRAY*

I.V.RAMAKRISHNAN, D.S.FUSSELL, and A.SILBERSCHATZ
Department of Computer Sciences
The University of Texas at Austin
Austin, Texas 78712

SUMMARY

An algorithm for multiplying two (N) x (N) matrices on a linear array requiring O(N) processors and $O(N^2)$ time steps was proposed in [1]. The processors used in the array were quite complex requiring dynamic control units and addressable memory. In this paper we present a systolic algorithm [2] for multiplying two (N) x (N) matrices on a linear array having the same processor and time complexity as the algorithm in [1]. However our algorithm uses simpler processing elements requiring no dynamic control units or addressable memory.

We begin our discussion with a description of our computation model. Let A, B and C be three (N) x (N) matrices where A and B are the two matrices to be multiplied and C is the result matrix. We will refer to the $(i,j)^{th}$ entry in A, B and C as a_{ij}, b_{ij} and c_{ij} respectively. The multiplication is performed on a linear array of identical processors numbered from 1 to 3N-2 as shown in figure 1. The array is driven by a single phase global clock.

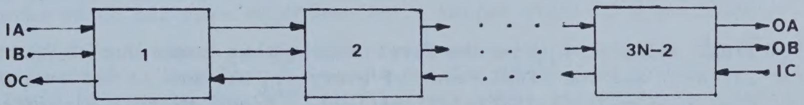


figure 1

The elements of A, B and C are pumped into the array through the input ports IA, IB and IC respectively. The elements of A and B and the result elements of C flow out of the array through the output ports OA, OB and OC respectively. A processor in the array consists of three input ports IP_A, IP_B, IP_C, three output ports OP_A, OP_B, OP_C, two shift registers S_B and S_C of lengths 1 and N-2 respectively, a multiplier and adder as shown in figure 2 where $\boxed{x}$ and $\boxed{+}$ denote multiplier and adder respectively.

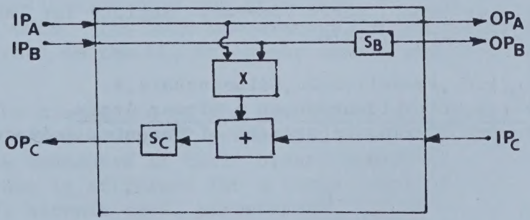


figure 2

---

*This research was supported in part by the National Science Foundation under Grant MCS-8104017 and ONR Contract N00014-80-k-0987

OP_A, OP_B of processor i is connected to IP_A, IP_B respectively of processor i+1 and OP_C is connected to IP_C of processor i-1. In every cycle (which is the same as a clock phase) a processor receives inputs a, b and c from its input ports IP_A, IP_B and IP_C respectively, computes $c+a*b$ and transmits a, b and $c+a*b$ to its output ports OP_A, OP_B and OP_C respectively through the appropriate shift registers. Thus if t is some cycle then a, b and $c+a*b$ appear at the output ports OP_A, OP_B and OP_C respectively in cycles t+1, t+2, and t+N-1 respectively.

We are in a position now to describe our algorithm. All the entries in C are initialized to 0 before being pumped into the array. Let t_s be the time at which c_{11} is fed into IC. For at least 3N-2 cycles prior to t_s pump 0 into IA. This ensures that the input port IP_A of every processor in the array is initialized to 0. Next perform the following steps:

1. Pump c_{ij} into IC at time $t_s+(i+j-2)*N+(i-1)$
2. Pump a_{ij} into IA at time $t_s+(2N-3)*(N-1)+(j-1)*N+(i-1)$
3. Pump b_{ij} into IB at time $t_s+(2N-5)*(N-1)+(N-j)+(i-1)*(N+1)$
4. For all times less than $t_s+(2N-3)*(N-1)$ (which is the time when a_{11} is pumped into IA) and for all times greater than $t_s+(N-1)*(3N-2)$ (which is the time at which a_{NN} is pumped into IA) pump 0 into IA.

The final value of c_{11} is the first result that comes out of OC at time $t_s+(3N-2)*(N-1)$ and the final value of every c_{ij} appears in the output port OC at time $t_s+(3N-2)*(N-1)+(i+j-2)*N+(i-1)$. Proof of this algorithm and its generalization to non-square matrices is given in [3].

REFERENCES

[1] Frank,G.A.,Greenawalt,E.M.,Kulkarni,A.V.
A Systolic Processor for Signal Processing.
In Proceedings of NCC. , 1982.

[2] Kung,H.T.,Leiserson,C.E.
Systolic Arrays (for VLSI).
In Sparse Matrix Proc., pages 256-282. Society for Industrial and Applied Mathematics, 1978.

[3] Ramakrishnan,I.V.,Fussell,D.S.,Silberschatz,A.
Systolic Matrix Multiplication on a Linear Array.
Technical Report TR-210, Department of Computer Sciences, U.T.Austin, 1982.

A NEW LAYOUT FOR CCC AND A NEW VLSI STRUCTURE - CCC AND MESH

XIALONG JIN*
Shanghai Institute of Railway Technology
Shanghai, China

ABSTRACT

In this paper a new layout for the cube-connected-cycle (CCC) is presented. With some additional links in the new layout the new structure (CCCM) can function either as a CCC or a mesh. For "small" values of N (up to 2^{11}) the chip area of these layouts is much smaller than that of other CCC and shuffle-exchange graphs. The asymptotic complexity of area is of $O(N^2/\log^2 N)$ as with other known results, but the coefficient on the $N^2/\log^2 N$ term is smaller than previous results. The CCCM is powerful for the execution of several classes of algorithms and is flexible for those computational intensive problems which require several phases of computation. The CCCM can be used as a general purpose parallel processor or a VLSI chip.

1. INTRODUCTION

A number of interconnected networks for VLSI implementation have been studied recently. Among them are mesh, hexagonally connected mesh [1], binary tree [2], shuffle-exchange graph [3], and cube-connected cycle (CCC) [4]. Each network is efficient for a particular class of algorithms. The design goal of general purpose VLSI chips is to find new interconnected networks which are more efficient for a larger class of algorithms.

Recently Kleitman et al. [5] presented a new layout for the shuffle-exchange graph. This layout, as the one for the CCC of Preparata and Vuillemin [4], meets the $O(N^2/\log^2 N)$ lower bound for the area, where N is the number of processing modules. However, from a practical point of view, we are interested in improving the coefficient and especially in getting a better layout for "small" values of N, which are more practical for implementations as VLSI chips.

The contribution of this paper is twofold. A new layout of CCC is given with asymptotic complexity of area of $O(N^2/\log^2 N)$, but with a coefficient smaller than previously known. For "small" values of N (up to 2^{11}) the area is much smaller than in previously proposed layouts.

The other contribution is that with some additional links this layout permits the realization of a new structure, by combining a CCC and a mesh, which we call CCCM. The CCCM can execute the algorithms, which are executable either on the CCC or on the mesh, resulting in a more powerful structure.

Before the discussion of the new layout, we briefly recall the CCC. A K-cube is a network with 2^K processors, numbered from 0 to 2^K-1. Two processors are connected if their binary numbering differs in exactly one bit. The K-cube is efficient for a large class of algorithms, such as FFT, bitonic merge, bitonic sort, permutation, convolution and so on, but its K connections per processor make it not feasible for VLSI layout.

The CCC is a modification of the K-cube. It can emulate the performance of the K-cube with respect to algorithms mentioned above without

---

[*] This work was done partly while the author was visiting the Coordinated Science Laboratory, University of Illinois at Urbana-Champaign.

increasing the processing time significantly, but the number of connections per processor is reduced to 3, which makes it feasible as a VLSI chip.

When the number of processors is $N = 2^K$, we can select a least number r such that $r + 2^r \geqslant K$. A CCC is a (K-r)-cube in which each vertex is a cycle with 2^r processors. Here the K-bit address m of each processor is expressed as a pair (ℓ,p) of integers represented with (K-r) and r bits respectively such that $\ell \cdot 2^r + p = m$. Each processor has three ports: F, B and L, whose connections are

$$F(\ell,p) \text{ -- } B(\ell,(p+1) \bmod 2^r)$$

$$B(\ell,p) \text{ -- } F(\ell,(p-1) \bmod 2^r)$$

$$L(\ell,p) \text{ -- } L(\ell+(1-2\,\mathrm{bit}_p(\ell))2^p,p) \ .$$

If $\langle X_0,\ldots,X_{K-r-1}\rangle$ are the dimensions of the (K-r)-cube, all edges along dimension X_i are called collectively sheaf i, and link pairs of processors whose addresses are of the form $(\cdot,i)$.

Figure 2 represents the layout of CCC with K = 6, r = 2. Here each cycle has 4 processors and regarding each cycle as a vertex, these 2^4 vertices form a 4-cube. Moreover, we can extend this construction to the case where each cycle has h processors and 2^s cycles form a s-cube for a total of $N = h \cdot 2^s$ processors. For details of the CCC, see [4].

This paper is organized as follows: Section 2 introduces layout method 1, which leads to a very compact layout and facilitates the embedding of a mesh. Section 3 introduces two other layout methods which are better for the coefficient of the asymptotic complexity of area. Section 4 presents a new structure, the CCCM formed from CCC with some additional links. Section 5 compares our layouts with other known layouts.

2. LAYOUT METHOD 1

The main idea of this new layout comes from the Karnaugh Map. A Karnaugh Map is a way to represent a K-cube in the plane. The Karnaugh Map in Fig. 1 (a) represents a 4-cube, while Fig. 1 (b) is an alternative representation as a graph, whose edges describe cell-adjacencies. Here a small dot represents a processor (hereafter we shall refer to it as a vertex).

As mentioned earlier, a CCC is a (K-r)-cube if we consider a 2^r-cycle as a vertex. If in Fig. 1 (b) we regard each vertex as a 4-cycle, then we can easily get a layout of the CCC with N = 64, K = 4, r = 2 (Fig. 2). In Fig. 2, the shaded squares are cycles. The I/O lines must transverse the boundary of the chip, and each of them must reach a cycle. For the cycles in columns 1 and 4, the I/O lines do not enlarge the area, so we do not show them. But for columns 2 and 3, the I/O lines need additional area as shown. The area of this layout is remarkably reduced to one half of the area of the corresponding layout in [4]. The layout of Fig. 2 is the basis for layouts with larger N; we call this connection pattern, i.e., 16 cycles connected like a 16-cell Karnaugh Map, a basic block.

For brevity, we denote by C_s the CCC with $N = s \cdot 2^s$ processors. The defined basic block is the layout of C_4.

We discuss next the layout of C_5. We place 16 cycles, each of which is a 5 × 1 rectangle with 5 processors, into 4 columns and connect them into a basic block. Next we juxtapose another identical structure, using a horizontal line between these two basic blocks as the line of symmetry. We complete the layout by connecting the fifth (unconnected) processor of each cycle of one basic block with its symmetric one in the other basic block, and create a channel between columns 2 and 3 to insert I/O lines. The resultant layout is shown in Fig. 3, where the shaded crosshatched rectangles are again the cycles. To emphasize the connection between the two basic blocks, we use bold lines to express these connecting lines, which consist of sheaf 4. There are 16 vertical lines in sheaf 4, 4 lines per column. Notice that sheafs 3, 2, 1 and 0 have been realized within the basic block.

Two comments are in order with regard to Fig. 3. The fifth processors of cycles in each column are located in the same vertical position. The purpose is to connect them into a mesh (see Section 4 for details). For the same reason, we use repeaters. In Fig. 3, each horizontal bold line links an open circle and a dot together. The open circle is the repeater of the dot. We can consider a processor, its repeater and the connection between them together as a whole processor with a larger area. Although this kind of processor occupies a larger area, the additional area is of no use. It is convenient to communicate this kind of processor with other processors without increasing any area. If the size of a processor is larger than the size of a wire, then we can put more than one wire in one side of the processor, making the repeater not necessary.

For the convenience of discussion, we refer to the vertical lines in a column, by which two C_i's are connected to form a C_{i+1}, as a subsheaf of sheaf i collectively. In Fig. 3, the four vertical lines in each column connecting two basic blocks are a subsheaf of sheaf 4.

Concerning the subsheafs of sheaf 4 in Fig. 3, we can see that symmetrically all the columns are the same. Once we have drawn the subsheaf in a column, say Column 2, we can get the corresponding subsheafs in the other columns in the following way:

1. Mirror the subsheafs of Column 2 in column 1 with the vertical symmetrical axis between these two columns.

2. Mirror the subsheafs of Columns 1 and 2 in Columns 4 and 3 respectively with the vertical symmetrical axis between Columns 2 and 3.

In general, the layout of C_s is like Fig. 3. Each sheaf i, $4 \leqslant i \leqslant s-1$, has 2^{s-1} vertical lines, i.e. each column having 2^{s-3} lines which consist of 2^{s-i-1} subsheafs and connect the cycles between basic blocks. Each subsheaf of sheaf i occupies a width of 2^{i-2}, so the length of a cycle should be

$$1 + \sum_{i=0}^{s-5} 2^{s-i-3} = 2^{s-3} + 3 .$$

We refer to the endpoints of the vertical line in sheaf i as v_i, $4 \leqslant i \leqslant s-1$. If the number s of the vertices in a cycle is even, then align them vertically in pairs. The purpose is to form a mesh with additional links and to make the chip perform as a mesh and as a CCC. The layout of C_s can be formed by connecting pairs of v_{s-1} in two layouts of C_{s-1} recursively. That is to say, for example, we connect pairs of v_4 in two layouts of C_4 to get the layout of C_5, we connect pairs of v_5 in two layouts of C_5 to get the layout of C_6, and so on.

For the description of a brief layout procedure, we distinguish two longer edges in a cycle; one belongs to A, another belongs to B. When we label the four cycles in a column of the basic block from bottom to top, the lower edges in cycles 1 and 3 and the upper edges in cycles 2 and 4 belong to A. The other 4 edges belong to B.

Layout Procedure

S1. Put 2^{s-4} basic blocks in vertical direction. Each cycle is a $(2^{s-2}-3) \times 1$ rectangle, and the width between Columns 2 and 3 is $2^{s-2}+1$.

S2. Put v_i, $i = s-1, s-3, \ldots, 5$ (s = even) or 4 (s = odd), in the A edge of each cycle with abscissa $x_i = 2^{s-2} - 2^{i-2}$. Put v_i, $i = s-2, s-4, \ldots$ 4 (s = even) or 5 (s = odd), in the B edge of each cycle with abscissa $x_i = x_{i+1}$.

S3. Connect pairs of v_{s-1} in two layouts of C_{s-1} recursively.

S4. Mirror the vertical lines in Column 1 with the vertical symmetry axis between these two columns, layout the I/O lines, and mirror the vertical lines in the right half of the layout.

For the discussion of layout parameters, we use the following notations.

a_s, b_s - sizes of the layout of C_s in x and y direction respectively.

A_s - area of the layout of C_s.

w_s - total width occupied by lines of sheaf i, $4 \leqslant i \leqslant s-1$, in a column.

ℓ_s - the length of a cycle.

Consider the sheaf i, $4 \leqslant i \leqslant s-1$. One subsheaf of sheaf i occupies 2^{i-2} width, so

$$w_s = \sum_{i=0}^{s-5} 2^{s-i-3} = 2^{s-2} - 4$$

and

$$\begin{aligned} \ell_s &= w_s + 1 = 2^{s-2} - 3 \\ a_s &= 4\ell_s + 7 + 2(2^{s-2} - 1) = 5 \cdot 2^{s-2} - 7 \\ b_s &= 3 \cdot 2^{s-2} - 1 \end{aligned}$$

so

$$A_s = (a_s+1)(b_s+1) = \frac{15}{16} 2^{2s} - \frac{9}{2} 2^s .$$

since $s \simeq \log(N/\log n)$, the chip size is about

$$\frac{15}{16} \left(\frac{N}{\log N}\right)^2 - \frac{9}{2} \frac{N}{\log N}$$

(in this paper all the logarithms are base 2).

For the case of $N = h \cdot 2^s$, $h > s$, we evaluate the cycle length of C_s and then increase it by $\lfloor \frac{1}{2}(h-s) \rfloor$. The layout is almost the same and the area increases by $3 \cdot 2^s \lfloor \frac{1}{2}(h-s) \rfloor$.

We can regard the case of $N = 2^K$ as the one of $N = h2^s$, where $h = 2^r$, $s = K-r$, and r is the smallest integer for which $r + 2^r \geqslant K$ (see [4]).

In both cases above, it is advantageous to layout the vertices in a mesh form.

3. OTHER LAYOUT METHODS

Combining Preparata and Vuillemin's idea in [4] and Karnaugh Map, we develop another layout method. This is shown in Fig. 4. It can be constructed recursively. Using the same notation, we have

$$\ell_s = \begin{cases} \frac{1}{3}(2^{s-1} - 4) + \frac{s-5}{2} & s = \text{odd} \\ \frac{1}{3}(2^{s-1} - 8) + \frac{s-4}{2} & s = \text{even} \end{cases}$$

and

$$A_s = \begin{cases} \frac{41}{64}\, 2^{2s} + \frac{3}{2}\, s \cdot 2^s - \frac{5}{2}\, 2^s & s = \text{odd} \\ \frac{41}{64}\, 2^{2s} + \frac{3}{2}\, s \cdot 2^s - 2^s & s = \text{even} \end{cases}$$

which is about

$$A_s = \begin{cases} \frac{41}{64} \left(\frac{N}{\log N}\right)^2 + \frac{3}{2} N - \frac{5}{2} \frac{N}{\log N} & s = \text{odd} \\ \frac{41}{64} \left(\frac{N}{\log N}\right)^2 + \frac{3}{2} N - \frac{N}{\log N} & s = \text{even} \end{cases}$$

so

$$A_s = O\left(\frac{N^2}{\log^2 N}\right).$$

The difference with [4] and Section 2 is in the coefficient. However there is no advantage in forming mesh + CCC for this layout method.

Moreover, we can change the basic block in a form as shown in Fig. 5. This allows easy access for the I/O lines. Figure 5 is a layout for C_s with $N = 5 \times 2^5$ processors. The area of this layout is about

$$\frac{2}{3} \left(\frac{N}{\log N}\right)^2.$$

4. CCC + MESH

With some additional links, the new layout of CCC contains a mesh having the same number of processors as the CCC. It is shown in Fig. 6, in which the dashed lines denote the additional links. Some processors have switches that can open or close the links. In this way this structure can function either as a CCC or as a mesh. Comparing Fig. 6 with Fig. 2, we see only a few modifications (shown by bold lines). We call this structure a Cube-Connected-Cycle Mesh or CCCM. The CCC is capable of executing a large class of algorithms efficiently [4] while the mesh is efficient for another class of algorithms, such as those arising in matrix and graph problems (Kung [1]). Combining these two powerful structures into one, the CCCM makes it possible to solve many problems which have to be solved with several phases and different phases may require different interconnections. For these problems, the CCCM provides a flexible pattern. If a problem needs several phases, either the CCC or mesh is efficient for each phase, then we can efficiently solve this problem with a single CCCM, using only one chip and avoiding data movement between chips which usually is a bottleneck.

5. COMPARISON WITH OTHER RESULTS

As mentioned in [5], a CCC has the same computing power as a shuffle-exchange graph. In Table 1 we compare the asymptotical complexity of area of these layouts with the best known results of shuffle-exchange graph [5], [6] and the former layout of CCC [4].

All these layouts have $O(N^2/\log^2 N)$ area, but our layouts have smaller coefficients. For "small" values of N which are practical for VLSI chip implementation, we list the best results of these layouts (Table 2) to compare the chip size. From this table we see that our layouts require less area than others.

ACKNOWLEDGEMENTS

I would like to thank Professor F. P. Preparata for many helpful discussions and G. Bilardi for his comments on the presentation of these results.

REFERENCES

[1] H. T. Kung, "Let's design algorithms for VLSI systems," Proc. Conference on Very Large Scale Integration: Architecture, Design, Fabrication, California Institute of Technology, January 1979.

[2] S. Browning, "Algorithms for the tree machine," in Introduction to VLSI Systems,

[3] H. S. Stone, "Parallel processing with the perfect shuffle," IEEE Transactions on Computers, C-20, February 1971.

[4] F. P. Preparata and J. Vuillemin, "The cube-connected-cycles: A versatile network for parallel computation," Proc. 20th Annual Symposium on Foundations of Computer Science, October 1979.

[5] D. Kleitman, F. T. Leighton, M. Lepley and G. L. Miller, "New layouts for the shuffle-exchange graph," (extended abstract), Proc. 13th Annual ACM Symposium on the Theory of Computing, Milwaukee, WI, May 1981.

[6] F. T. Leighton and G. L. Miller, "Optimal layouts for 'small' shuffle-exchange graphs," (abstract), Applied Mathematics Department, MIT, 1981.

TABLE 1

	Asymptotic Complexity	Comments
CCC in [4]	$\frac{4}{3}\left(\frac{N}{\log N}\right)^2$	
shuffle-exchange graph [5]	$C\left(\frac{N}{\log N}\right)^2$	C at least 2
new layout 1	$\frac{15}{16}\left(\frac{N}{\log N}\right)^2$	can be extended to Mesh + CCC
new layout 2	$\frac{41}{64}\left(\frac{N}{\log N}\right)^2$	
new layout 3	$\frac{2}{3}\left(\frac{N}{\log N}\right)^2$	using another basic block

TABLE 2

	N						
	32	64	128	256	512	1024	2024
CCC in [4]	128	384	448	1600	6016	22528	89600
Shuffle-exchange graph [5], [6]	63	198	684				
new layout 1	48	168	240	912	3552	14784	61056
new layout 2	48	168	240	816	3072	11328	44416
new layout 3	48	192	256	896	2816	12032	44032

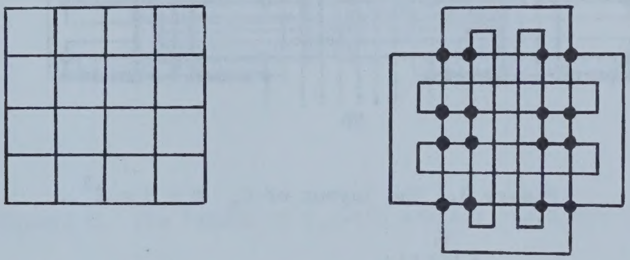


Figure 1. A 16-cell Karnaugh map.

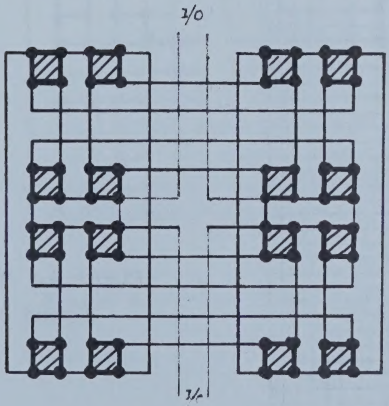


Figure 2. The layout of C_4, N = 64 A = 12 × 14 = 168.

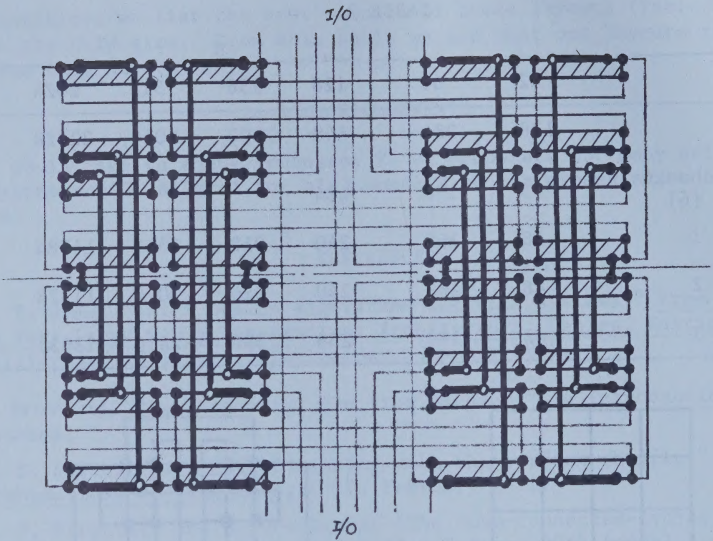


Figure 3. The layout of C_5 $N = 5 \times 2^5$.

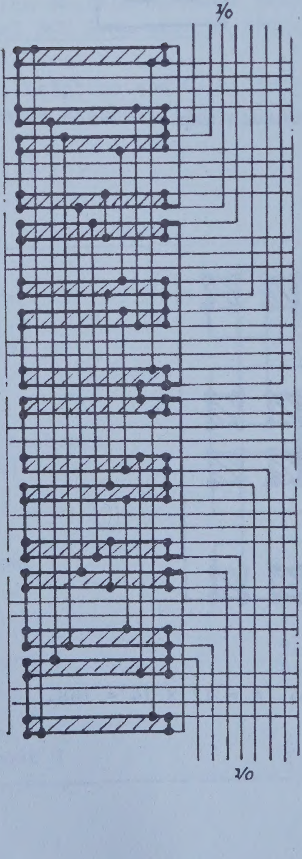


Figure 4. The layout of second column of C_6.

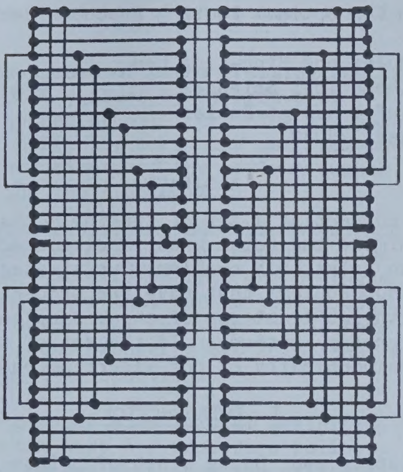


Figure 5. The layout of C_5 with another basic block

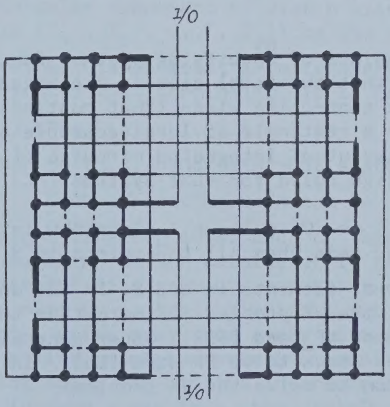


Figure 6. CCC + Mesh = CCCM

An Optimal Algorithm for Optimal Routing Around a Rectangle[†]

Teofilo F. Gonzalez and Sing-Ling Lee
Programs in Mathematical Sciences
The University of Texas at Dallas
Richardson, Texas 75080

ABSTRACT

The problem of connecting a set of terminals that lie on the sides of a rectangle to minimize the total area is discussed. We present an O(n log n) algorithm to solve this problem. Our strategy is to reduce the problem to several subproblems that have the property that no matter what instance we start with, at least one of these subproblems can be solved optimally by the greedy method. If the set of terminals is initially sorted, then the time complexity of our algorithm reduces to O(n α (n)).

I. INTRODUCTION

Let T be a rectangle and S be a set of points on the sides of T. Each point in set S has to be connected by a wire to exactly one point in set S. The wires can be assigned to two different layers. It is assumed that all wires in one layer are parallel to the x-axis and all wires in the other layer are parallel to the y-axis. Wires in different layers can be connected at any given point z by a wire perpendicular to the layers if both wires cross point z in their respective layers. Every pair of distinct and parallel wires must be at least λ units apart and every wire must be at least λ units from each side of T, except in the region where the wire joins the point in S it connects. Also, no wire is allowed inside T in any of the layers.

Problem WR consists of specifying paths for all the connecting wires in such a way that the total area is minimized. That is, to place T together with all the connecting wires (that must satisfy the restrictions imposed above) inside a rectangle of least possible area. This problem has applications in the layout of integrated circuits ([L] and [R]) and conforms to a set of design rules for VLSI systems [MC].

In [HS] and [GLL] an O(n log n) algorithm is presented to solve the WR problem for the case when all the points in S lie in one side of T, where n is the number of elements in set S. In [L] an O(n^3) algorithm is presented to solve the WR problem. If more than two layers are allowed, then the problem becomes NP_hard [SBR]. Other generalizations of the WR problem have also been shown to be NP-hard [La]. In this paper we present an O(n log n) algorithm to solve the WR problem.

Initially our algorithm follows the steps of LaPaugh's algorithm [L]. These steps are the divide-and-conquer and the final assignment of the direction for the connecting paths of all the local terminals (a terminal is said to be local if it is to be connected to another terminal that is located on the same side or on an adjacent side of T). After these steps, it is only required to solve a restricted version of the WR problem. In this restricted version the only non-local terminals appear on the top and bottom sides of T. It is at this point that our algorithm will differ from the one given in [L].

[†] This work was supported in part by ORG grant A6495-51

First we show the existence of an optimal solution that satisfies the following properties:

a) Balanced. The number of paths crossing two predefined points on the top side of T differs by at most one.

and b) The maximum number of paths crossing any point on the top side of T will not differ by more than one from the minimum, taken over all feasible solutions, of the maximum number of paths crossing any point on the top side of T.

Two suboptimal solutions satisfying conditions (a) and (b) are defined. Both of these solutions can be easily generated. We then show that there is an optimal solution that differs from one of these suboptimal solutions by a set of connecting paths that can be easily characterized. Our algorithm generates these suboptimal solutions and interchanges the direction of several sets of paths producing two solutions. At least one of these solutions will be an optimal solution.

In section II we present some initial definitions and the steps from LaPaugh's [L] algorithm that our procedure follows. Then in section III we present a series of lemmas that show the existence of an optimal solution that can be generated by the greedy method. The final algorithm and complexity issues relating to the WR problem are discussed in section IV.

II. Definitions and Problem Reductions

In this section we redefine the WR problem. Our algorithm can be easily explained under this new definition. We also define some terms and present the steps of LaPaugh's algorithm that our procedure follows.

Let T be a rectangular component of size h by w (height by width). There are 2n terminals (T_1 , T_2 , ... , T_{2n}) on its sides. It is assumed that every pair of terminals is at least λ units apart. The function $\underline{C(i)}$, for $1 \leq i \leq 2n$, indicates that terminal T_i is to be connected to terminal $T_{C(i)}$. If $C(i) = j$ then $C(j) = i$, i.e., C is a symmetric function. T_i is to be connected to $T_{C(i)}$ by a path that starts at terminal T_i and ends at terminal $T_{C(i)}$. Each connecting path can be partitioned into a finite number of straight line segments. Each of these line segments must lie on the same plane as T, be on the outside of T and be parallel to a side of T. Perpendicular line segments can intersect at any point, but parallel line segments must be at least λ units apart. Also, all line segments must be at least λ units away from every side of rectangle T except in the vicinity where a line segment connects a terminal. The WR problem consists of specifying paths for all the interconnections subject to the rules mentioned above in such a way that the total area is minimized, i.e., place the component together with all the interconnecting paths inside a rectangle of least possible area.

Label the sides of the component (in the obvious way) left, top, right and bottom. Starting in the left-bottom corner of T, traverse the sides of the rectangle clockwise. The ith corner to be visited is labeled S_{i-1}. Assume that the ith terminal visited is terminal T_i . The close interval $[\underline{x} , \underline{y}]$, where x and y are the corners of T or the terminals T_i, consists of all the points on the sides of T that are visited while traversing the sides of T in the clockwise direction starting at point x and ending at point y. Parentheses are used instead of square brackets when it is desired to specify an open interval. We use $[S_0 , S_1]$, $[S_1$,

$S_2]$, $[S_2 , S_3]$ and $[S_3 , S_0]$ to represent the left, top, right and bottom sides of T, respectively. Terminal T_i is said to belong to side ℓ, *S(i)* = ℓ , if T_i is located in $[S_\ell , S_{(\ell+1)\bmod(4)}]$.

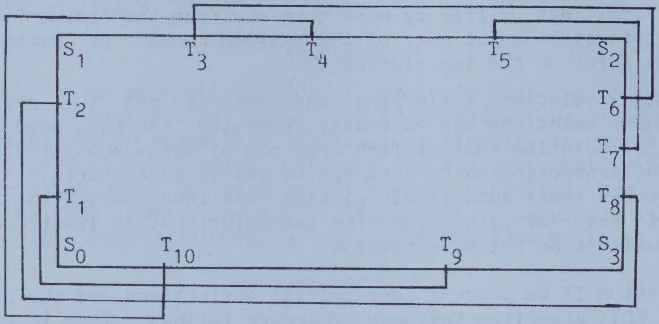


figure 1.1

Set $D = \{d_1 , d_2 ,\dots, d_k\}$ is said to be an *assignment of directions* if the cardinality of set D is n
and $\{ d_i , C(d_i) \mid 1 \le i \le n \} = \{1,2,\dots,2n\}$.

D is said to be a *partial assignment* if the cardinality of set D is $\le$ n,

$\{ d_i , C(d_i) \mid 1 \le i \le k \} \subseteq \{1 , 2 ,\dots, 2n\}$,

and $|\{ d_i , C(d_i) \mid 1 \le i \le k \}| = 2 * k$.

An assignment D indicates the direction of the connecting path for all terminals. The connection of terminal T_i, for any i in D, to terminal $T_{C(i)}$ starts at terminal T_i moving perpendicular to side S(i), it then continues moving in the clockwise direction with respect to T until it can be joined to a line segment (all of it on the outside of T) perpendicular to S(C(i)) that starts at terminal $T_{C(i)}$. In a partial assignment, the direction of some connecting paths might not be specified. The assignment for the layout given by figure 1.1 is {3 , 5 , 8 , 9 , 10}. For any $\ell \in$ D, we say that the connecting path for ℓ as given by D *crosses* point z if $z \in [\ell , C(\ell)]$.

For any assignment (or partial assignment) D we define the *layer function* H_D for $x,y \in \{T_1,T_2,\dots, T_{2n}\}$ (_) $\{S_0,S_1,S_2,S_3\}$ as follows:

$H_D(x,y) = \max\{\text{number of paths given by D that cross point } z \mid z \in [x,y]\}$.

Let D be the assignment for the layout shown in figure 1.1. For assignment D we have that $H_D(S_0 , S_1)$ is 3, $H_D(T_5 , T_5)$ is 2 and $H_D(S_2 , S_3)$ is 2. We shall refer to $H_D(x , y)$ as the *height of the interval* [*x* , *y*] *in T for assignment D*.

The next two lemmas establish that the WR problem reduces to the problem of finding an assignment D with least

$$(H_D(S_1 , S_2) + (H_D(S_3 , S_0)) * (H_D(S_0 , S_1) + (H_D(S_2 , S_3))$$

and then in O(n log n) time (O(n) time if the set of terminals is initially sorted) one can construct a final layout for the connecting paths.

Lemma 2.1: For every assignment D, there is a rectangle Q of size h_Q by w_Q, where

$h_Q = h + (H_D(S_1,S_2)+H_D(S_3,S_0)) * \lambda$ and $w_Q = w + (H_D(S_0,S_1)+H_D(S_2,S_3)) * \lambda$

with the property that rectangle T together with the interconnecting paths defined by D can be made to fit inside Q.

Proof: The proof appears in [L]. []

Lemma 2.2: A final layout with the area given by lemma 2.1 can be obtained in O(n log n) time (O(n) time if the set of terminals is initially sorted) for any assignment D.

Proof: The proof of this lemma appears in [L]. The procedure that constructs the final layout, uses as a subalgorithm the procedures given in [GLL] and [HJ]. []

Terminal T_i is said to be a global terminal if $|S(i) - S(C(i))| = 2$, i.e., terminal T_i is global if it is to be connected to a terminal located on the opposite side of the rectangle. Terminal T_i is said to be local otherwise, i.e., if it is to be connected to a terminal located on the same side or on an adjacent side of the rectangle T. The problem shown in figure 1.1 has T_2, T_8, T_4 and T_{10} as the only global terminals. For assignment D we define the function A(D) as

$(h + (H_D(S_1, S_2) + H_D(S_3, S_0)) * \lambda) * (w + (H_D(S_0, S_1) + H_D(S_2, S_3)) * \lambda)$

i.e., the total area required by the layout of T together with all the interconnections specified by D.

Definition 2.1: D'

Let D' be the partial assignment in which all the local terminals are connected by paths crossing the least number of corners of T. []

Lemma 2.3: There is an optimal assignment, D, such that D' $\subseteq$ D.

Proof: The proof appears in [L]. []

Lemma 2.3 shows that given any instance of the WR problem it is possible to find an optimal solution in which all local terminals are connected by paths that cross at most one corner of T. The WR problem has been reduced to the problem of finding the direction for the connecting paths of all global terminals in the presence of the partial assignment D'. The next lemma partitions the WR problem into two separate problems: the problem of finding an optimal solution to the WR problem in which all global terminals appear in the top and bottom sides of T and the one in which all global terminals appear in the left and right sides of T. In both these subproblems, local terminals are connected by the paths specified in D'.

Lemma 2.4: D is an optimal assignment if and only if both

$h_Q = h + (H_D(S_1,S_2)+H_D(S_3,S_0)) * \lambda$ and $w_Q = w + (H_D(S_0,S_1)+H_D(S_2,S_3)) * \lambda$

are optimal.

Proof: The proof of this lemma appears in [L]. []

III. The restricted problem

In this section we show that given any instance of the restricted WR problem, it is always possible to obtain an optimal solution by solving one of its subproblems with the greedy method.

From now on we shall restrict our attention to the solution of the WR problem in which all global terminals are located on the top and bottom sides of T and all local terminals are connected by the paths specified in D'. Note that the second subproblem mentioned in the previous section can be transformed to this one by rotating the rectangle 90 degrees. If the number of global terminals located on the top side of T is zero, then D' is an optimal assignment (lemma 2.3). In what follows we assume there is at least one global terminal located on the top side of T.

First the points T_α and T_β will be defined. It will then be shown the existence of an optimal solution in which the global terminals located in the interval $[S_1 , T_\alpha)$ and $(T_\beta , S_2]$ are connected as in assignment D" (def 3.2).

Definition 3.1: α and β.

Let

$L = D' \cup$ {all global terminals are connected by a path crossing the left side of T};

$\alpha = \max\{ k \mid H_L(T_k , T_k) = H_L(S_1 , S_2)$ and $T_k \in [S_1 , S_2]\}$;

$R = D' \cup$ {all global terminals are connected by a path crossing the right side of T};

and $\beta = \min\{ k \mid H_R(T_k , T_k) = H_R(S_1 , S_2)$ and $T_k \in [S_1 , S_2]\}$. []

Definition 3.2: D"

Let $D'' = D' \cup$ {all global terminals located in the interval $[S_1 , T_\alpha)$ connected by a path crossing the left side of T} $\cup$ {all global terminals located in the interval $(T_\beta , S_2]$ connected by a path crossing the right side of T} []

Definition 3.3: α' and β'

Let $\alpha' = H_{D''}(T_\alpha , T_\alpha)$ and $\beta' = H_{D''}(T_\beta , T_\beta)$. []

It will now be shown that there is an optimal assignment such that D" is a subset of it.

Lemma 3.1: Let D" be as defined above. There is an optimal assignment, D, such that $D'' \subseteq D$.

Proof: The proof of this lemma appears in [GL]. []

Definition 3.4: t

Let t be the number of global terminals in the interval $[T_\alpha, T_\beta]$. []

If t = 0 then D" is a complete assignment, i.e., we have specified the direction for the connection of all terminals, and by lemma 3.1 we conclude that D" is an optimal assignment. In what follows we shall assume that $t > 0$.

A lower bound for the height on the top side of T for any optimal solution satisfying the conditions of lemma 3.1 is given by $\triangle$ which is defined below.

Definition 3.5: $\triangle$

$\triangle = \lceil (\alpha' + \beta' + t) / 2 \rceil$ []

In what follows we show that there is an optimal solution satisfying lemma 3.1 whose height on the top side of T is $\triangle$ or $\triangle + 1$ and with the property that the number of paths crossing at points T_α and T_β differs by at most one.

Lemma 3.2: There is an optimal solution, D, such that

(a) $D'' \subseteq D$;

(b) $\left| H_D(T_\alpha , T_\alpha) - H_D(T_\beta , T_\beta) \right| \leq 1$

and (c) $\Delta \leq H_D(S_1 , S_2) \leq \Delta + 1$.

Proof: The proof of this lemma appears in [GL]. []

There are instances for which there is no optimal solution satisfying (a) and (b) of lemma 3.2 and whose height on the top side of T is Δ. Also, there are instances that do not have an optimal solution satisfying (a) and (b) of lemma 3.2 and whose height on the top side of T is $\Delta + 1$. When $\alpha' + \beta' + t$ is odd, one can show that there always exists an optimal solution whose height on the top side of T is Δ. The proof of this fact is similar to the one of lemma 3.2. In the next two lemmas we do not take advantage of this fact, however it will be used by the final algorithm.

We now define assignments E_1 and E_2. These assignments have the property that there is an optimal assignment that differs from them by a set of connecting paths that can be easily characterized.

Definition 3.6: E_1 and E_2

Let

$M = \{ \ell \mid T_\ell$ is a global terminal located on the bottom side of T and $\ell , C(\ell) \notin D''\}$;

$M_1 = \{ \ell \mid \ell$ is in M, ℓ is the ith largest value in M and $i \leq \lceil (\alpha' + \beta' + t) / 2 \rceil - \alpha'\}$;

$M_2 = \{ \ell \mid \ell$ is in M, ℓ is the ith largest value in M and $i \leq \lfloor (\alpha' + \beta' + t) / 2 \rfloor - \alpha' \}$;

$E_1 = D'' \cup \{ \ell \mid \ell \in M_1\} \cup \{ C(\ell) \mid \ell \in (M - M_1) \}$;

and $E_2 = D'' \cup \{ \ell \mid \ell \in M_2\} \cup \{ C(\ell) \mid \ell \in (M - M_2)\}$. []

We should point out that if $\alpha' + \beta' + t$ is even then $E_1 = E_2$. However, if this is not the case, then E_1 and E_2 will differ only on the direction of one connecting path.

In what follows we define some terms for assignment E which can be either E_1 or E_2. When we make use of the terms defined this way, it will be explicitly indicated which of E_1 or E_2 was used in the definition.

Definition 3.7: R , T'_ℓ and T''_ℓ

Let $R = H_E(S_1 , S_2) - \Delta$.

For $\ell = 1 , 2 , \ldots , R$, let T'_ℓ (T''_ℓ) represent the terminal located in the interval $[S_1 , S_2]$ with largest (smallest) index whose height is $\Delta + \ell$. Let T'_0 (T''_0) represent T_β (T_α). []

We say that assignment D *conforms* to E_1 (or E_2) if $H_D(T_\alpha , T_\alpha) = H_{E_1}(T_\alpha , T_\alpha)$ (or $H_D(T_\alpha , T_\alpha) = H_{E_2}(T_\alpha , T_\alpha)$).

It is required to define both E_1 and E_2 because there are instances for which there is no optimal solution that conforms to E_1. The same is also true for E_2. Note that this happens only when $\alpha' + \beta' + t$ is odd.

Let D be an optimal assignment that satisfies the conditions of lemma 3.2 and let $E \in \{E_1 , E_2\}$ conform to D. A connecting path is said to be of *type I* (*type II*) if it crosses the right (left) side of T in D but not in E.

Lemma 3.3: There is an optimal solution D such that $E \in \{E_1 , E_2\}$ conforms to it and

a) $D'' \subseteq D$;

b) $\left| H_D(T_\alpha , T_\alpha) - H_D(T_\beta , T_\beta) \right| \le 1$;

c) $\Delta \le H_D(S_1 , S_2) \le \Delta + 1$;

d) all type I (type II) paths connect a terminal located in the interval $[T'_R , T_\beta]$ ($[T_\alpha , T''_R]$)

and e) The number of type I and type II paths is

$\lceil R / 2 \rceil$ if $H_D(S_1,S_2) = \Delta$ and $\lfloor R / 2 \rfloor$ otherwise.

Proof: The proof of this lemma appears in [GL]. []

Sets of terminals will be defined and subsets of them will be labeled A, A', B and B'. This sets will be used in lemma 3.4 where it will be shown that there is always an optimal solution that differs from E_1 or E_2 by the set of paths that connect the terminals labeled A and A' or B and B'. Consequently in order to construct an optimal assignment it is only required to construct E_1 and E_2 and then interchange some set of connecting paths. One of the assignments obtained this way will be an optimal assignment.

Definition 3.8: Q_ℓ , Q'_ℓ , P_ℓ and P'_ℓ.

For $\ell = 1 , 2 , \ldots , R$, let Q_ℓ (Q'_ℓ) be the set of indices of the global terminals located in the interval $(T'_\ell , T_\beta]$ ($[T_\alpha , T''_\ell)$) if ℓ is odd and $[T'_\ell , T_\beta]$ ($[T_\alpha , T'_\ell]$) otherwise, such that these terminals are connected by a path that crosses the left (right) side of T in assignment E.

For $\ell = 1 , 2 , \ldots , R$, let P_ℓ (P'_ℓ) be the set of indices of the global terminals located in the interval $[T'_\ell , T_\beta]$ ($[T_\alpha , T''_\ell]$) if ℓ is odd and $(T'_\ell , T_\beta]$ ($[T_\alpha , T''_\ell)$) otherwise, such that these terminals are connected by a path that crosses the left (right) side of T in assignment E. []

Definition 3.9: A , A' , B and B' labeling.
We shall label terminals as follows:

```
A labeling:
  for $\ell = 1$ to R do
    Let $f = \lceil \ell / 2 \rceil - \lceil (\ell - 1) / 2 \rceil$
    Let $X = \{ C(i) \mid i \in P_\ell$ and i was not labeled when considering sets
        $P_1 , P_2 , \ldots , P_{\ell-1} \}$
    Let X' = the f elements of smallest value in set X
    Each index i such that $C(i) \in X'$ is labeled A.
  endfor
A' labeling:
    This labeling procedure is identical to the previous one, except for
        the sets $P_\ell$ being replaced by $P'_\ell$ and "smallest" is replaced by
        "largest".
B labeling:
  for $\ell = 1$ to R do
    Let $f = \lfloor \ell / 2 \rfloor - \lfloor (\ell - 1) / 2 \rfloor$
    Let $X = \{ C(i) \mid i \in Q_\ell$ and i was not labeled when considering sets
        $Q_1 , Q_2 , \ldots , Q_{\ell-1} \}$
    Let X' = the f elements in set X with smallest value.
    Each index i such that $C(i) \in X'$ is labeled B.
  endfor
B' labeling:
    This labeling procedure is identical to the previous one, except for
        the sets $Q_\ell$ being replaced by $Q'_\ell$ and "smallest" is replaced by
        "largest". []
```

The next lemma establishes that there is an optimal solution which can be obtained by starting from E_1 or E_2 and reversing the connecting path for the terminals labeled A and A' or B and B'.

Lemma 3.4: There is an optimal assignment D, such that $E \in \{E_1 , E_2\}$ conforms to it and

a) $D'' \subseteq D$;

b) $|H_D(T_\alpha , T_\alpha) - H_D(T_\beta , T_\beta)| \leq 1$;

c) All type I (type II) paths connect a terminal located in the interval $[T'_R , T_\beta]$ $([T_\alpha , T''_R])$;

d) The number of type I and type II paths in D is $\lceil R / 2 \rceil$ if $H_D(S_1, S_2) = \Delta$ and $\lfloor R / 2 \rfloor$ otherwise.

and e) if $H_D(S_1 , S_2) = \Delta$ then all the type I (type II) paths in D connecting a terminal in the interval $[T'_R , T_\beta]$ $([T_\alpha , T''_R])$ connect a terminal with label A (A'). if $H_D(S_1 , S_2) = \Delta + 1$ then all the type I (type II) paths in D connecting a terminal in the interval $[T'_R , T_\beta]$ $([T_\alpha , T''_R])$ connect a terminal with label B (B').

Proof: The proof of this lemma appears in [GL]. []

IV. Algorithm and Complexity Issues

In this section we present our algorithm to solve the WR problem. The algorithm is based on the lemmas presented in sections II and III. Our algorithm has worst case time complexity $O(n \log n)$ and $O(n \alpha(n))$ when the set of terminals is initially sorted. In the last part of this section we discuss lower bounds on the worst case time complexity of decision tree algorithms for the WR problem.

We now present the algorithm:

```
algorithm ROUTING
 Rename the set of terminals in such a way that when traversing T in the
   clockwise direction starting at point $S_0$, the terminals are visited
   in the order $T_1, T_2, \ldots, T_{2n}$;
 Label the terminals local and global following the definitions that
   appear after lemma 2.2;
 Construct D';                                   //def 2.1//
 Partition the problem into the following two subproblems:
   $P_1$ is the original problem after deleting all global terminals
          appearing in the left and right sides of T, and
   $P_2$ is the original problem after deleting all global terminals
          appearing in the top and bottom sides of T;
 $D_1 \leftarrow$ SOLVE( $P_1$ );
 $D_2 \leftarrow$ SOLVE( $P_2$ );
 Combine $D_1$ and $D_2$ into the final assignment D;
 Construct and output the layout for D using the proof of lemma 2.2;
end of algorithm ROUTING;

procedure SOLVE( P );
 Construct D' for P;                        //def 2.1//
 if there are no global terminals then return( D' ) endif;
 Compute $\alpha$ and $\beta$;                        //def 3.1//
 Construct D";                              //def 3.2//
 Compute t;                                 //def 3.4//
 if t = 0 then return( D" ) endif;
 Compute $\Delta$;                                //def 3.5//
 Construct $E_1$ and $E_2$;          //def 3.6//
 Compute R;                                 //def 3.7//
 Define $T'_1, \ldots, T'_R, T''_1, \ldots, T''_R$ for $E_1$ and $E_2$;   //def 3.7//
 Perform the A , A' , B and B' labelings for $E_1$ and $E_2$;  //def 3.9//
 if $\alpha' + \beta' + t$ is even then $D_1 \leftarrow$ MODIFY(A , A' , $E_1$);
                                   $D_2 \leftarrow$ MODIFY(B , B' , $E_1$);
                              else $D_1 \leftarrow$ MODIFY(A , A' , $E_1$);
                                   $D_2 \leftarrow$ MODIFY(A , A' , $E_2$);
 endif
 return( $D_1$ if $A(D_1) \leq A(D_2)$ and $D_2$ otherwise );
end of procedure

procedure MODIFY( L , L' , E )
 D $\leftarrow$ E except that the connecting paths for all terminals labeled  L
      and L' is reversed;
 return( D );
end of procedure
```

Theorem 4.1: Algorithm ROUTING solves the WR problem.
Proof: The proof is based on lemmas 2.1 , 2.2 , 2.3 and 3.4. []

Theorem 4.2: The time complexity of procedure ROUTING is $O(n \log n)$.
Proof: The proof of this lemma appears in [GL]. []

Theorem 4.3: The time complexity of procedure ROUTING is $O(n \alpha(n))$ when the set of terminals is initially sorted.
Proof: The proof of this lemma appears in [GL]. []

In [GLL] it was shown that $\Omega(n \log n)$ comparisons are required by any decision tree algorithm that solves the 1-dimensional WR problem. This result holds even when comparisons among linear functions are allowed. A similar result can also be proven for the case when the input to the 1-dimensional WR problem is restricted to terminals located at a distance of at least λ units from each other. Clearly, this result also holds for the WR problem. Hence, the worst case time complexity for the WR problem is $\Theta(n \log n)$.

V. REFERENCES

[AHU] Aho, A. , Hopcroft, J. and Ullman, J. , " *The Design and Analysis of Computer Algorithms* ", Addison-Wesley, 1975.

[HS] Hashimoto, A. and Stevens J. E., "Wire Routing by Optimizing Channel Assignment without large Apertures", Proceedings of the 8th IEEE Design Automation Conference, 1971, pp 155-169.

[L] LaPaugh, A. S., "A Polynomial Time Algorithm for Optimal Routing around a Rectangle", Proceedings of the 21st IEEE Foundations of Computer Science, 1980, pp 282-29

[La] LaPaugh, A. S., "Algorithms for Integrated Circuit Layout, An Analytic Approach", Ph.D. dissertation, Massachusetts Institute of Technology, 1980.

[GL] Gonzalez, T and Lee, S, "An O(n log n) Algorithm for Optimal Routing Around a Rectangle," Technical Report #116, Programs in Mathematical Sciences, The University of Texas at Dallas, Oct. 1982.

[GLL] Gupta, U. I., Lee D. T. and Leung J., "An Optimal Solution for the Channel-Assignment Problem", *IEEE Transactions on Computers*, Vol. C-28, No. 11, 1979, pp. 807-810.

[MC] Mead, C. and Conway, L., " *Introduction to VLSI Systems* ", Addison-Wesley, 1980.

[R] Rivest, R.L., "The PI (Placement and Interconnect) System", Proceedings of the 19th IEEE Design Automation Conference, pp 475-481.

[SBR] Sahni, S., Bhatt, A. and Raghavan, R., "The Complexity of Design Automation Problems", Technical Report 80-23, Department of Computer Science, The University of Minnesota, Dec 1980.

ANALYSIS OF HEURISTICS FOR PLA FOLDING

S.S.RAVI and ERROL L. LLOYD
Department of Computer Science
University of Pittsburgh
Pittsburgh, PA 15260

ABSTRACT

Folding is a technique by which pairs of input and output lines (or product terms) of a Programmable Logic Array (PLA) can be made to share the same physical line. If a PLA could be folded completely, its area would be reduced by 75%. An analytical model for the folding problem was introduced in [HNS81]. It has recently been shown [LVVS82] that the Optimal Folding problem is NP-Complete. In view of its practical importance, it is interesting to consider efficient heuristics for this problem. In this paper, we first obtain an upper bound on the performance of any heuristic which finds a maximal set of folding pairs. We use this result to analyze the performance of some variants of a general heuristic proposed in [HNS81]. While the problem of obtaining a constant worst case bound for any PLA is open, we present a heuristic which achieves a constant bound for three special classes of PLA's.

1. INTRODUCTION

Due to their regular structure, Programmable Logic Arrays (PLA's) constitute an important building block of VLSI systems. Since chip area is a critical factor in determining the yield of VLSI chips [MC80], it is necessary to investigate methods to reduce the area occupied by the building blocks of a VLSI system. One method to compact the area of a PLA is to combine its input and/or output lines (or product terms) so that two inputs or outputs (or two product terms) share a physical line. This technique is known as Folding [HNS81]. If all input/output lines and all product terms of a PLA could be folded, the decrease in the area of the PLA could be as much as 75%. An analytical formulation of the PLA folding problem has been proposed in [HNS81]. It has recently been shown [LVVS82] that the optimal folding problem (i.e., the problem of finding the largest set of folding pairs) is NP-Complete. In view of the practical importance of this problem, it is interesting to investigate efficient heuristics for it. In [HNS81], a general heuristic was proposed and it was pointed out that the heuristic produces maximal folding sets. The authors of [HNS81] also implemented a variant of the general heuristic and found that it performs well in practice. The fact that this heuristic does well in practice makes it interesting to study its worst case performance. In this paper, we obtain an upper bound on the worst case behavior of any heuristic which obtains a maximal set of folding pairs. We use this result to analyze the worst case behavior of some variants of the general heuristic proposed in [HNS81]. In all cases, we find that these heuristics exhibit unbounded worst case

---

[†] This research was partially supported by the National Science Foundation under Grant MCS-8103713.

behavior. While the problem of achieving a constant worst case bound for general PLA's is open, we show that for some special classes of PLA's, it is possible to achieve a constant worst case bound.

2. THE PLA FOLDING PROBLEM

In this section, we briefly review the analytical model for the PLA folding problem, introduced in [HNS81]. Our intent here is to present just enough material to enable the reader to follow the analysis in the later sections. We encourage the interested reader to consult [HNS81] for more details. We assume that the reader is familiar with the basic structure of a PLA (Chapter 3 of [MC80] provides an excellent treatment).

A PLA consists of two matrices of circuit elements called AND and OR planes. The output of the AND plane is the set of product terms which are suitably combined in the OR plane to produce the outputs. A schematic of a PLA with 3 inputs and 2 outputs is shown in Figure 1(a). In that figure, a connection (indicated by a dark circle) in the AND plane (OR plane) means that the corresponding input (product term) is present in the product term (output). A PLA can also be represented in the form of a boolean matrix, called its Personality Matrix. The Personality Matrix P of a PLA has m rows and n columns where m is the number of distinct product terms and n is the total number of inputs and outputs. We refer to the set of inputs and outputs together simply as lines. An element p_{ij} of the matrix P is 1 if line j (or its complement) is connected to product term i, and is 0 otherwise. Figure 1(b) shows the Personality Matrix of the PLA in Figure 1(a). Columns r and s of a Personality Matrix intersect if there is at least one row q such that $p_{qr} = p_{qs} = 1$.

Folding (or Splitting) a PLA consists of dividing a line into two electrically disconnected pieces so that two lines can be made to share the same physical column[†]. A folded version of the PLA of Figure 1(a) is shown in Figure 1(c). It is clear that two lines can be folded only if the columns of the Personality Matrix corresponding to these lines do not intersect (i.e., the two lines are not required in the same product term). A pair of lines (l_1, l_2) satisfying this condition is a Candidate Folding Pair.

Folding a pair of lines leads to a reordering of the product terms. Specifically, a folding pair (l_1, l_2) causes the product terms requiring l_1 to be above or below those requiring l_2. A candidate folding pair (l_1, l_2) gives rise to an Ordered Folding Pair (l_1, l_2) if the product terms requiring l_1 are above those requiring l_2. Thus every ordered folding pair induces a relation on the set of product terms. A set of disjoint folding pairs forms an Ordered Folding

---

[†] This is strictly "Column Folding". It is also possible to fold the rows (the product terms). Our results also apply to row folding. For the purposes of this analysis we make no distinction between input and output lines when locating folding pairs.

Set. The relation R induced by an ordered folding set F is the union of the relations induced by the members of F. An ordered folding set is Physically Realizable if the reflexive-transitive closure of the relation R induced by F is a partial order (i.e., the arrangement of the product terms does not lead to cyclic constraints on the position of any product term). From this formulation, it is clear that we can test in polynomial time whether an ordered folding set is physically realizable.

We can now define the Optimal Folding Problem (OFP): Given the Personality Matrix of a PLA, find an ordered realizable folding set of maximum cardinality. It has been shown recently [LVVS82] that OFP is NP-Complete. Thus we are justified in considering heuristics for this problem. A graph theoretic formulation of OFP introduced in [HNS81] is very helpful in analyzing this problem. We need a few additional definitions for this purpose. The Intersection Graph G(V,E) of a PLA with Personality Matrix P is an undirected graph, defined as follows:

$$V = \{ l_1, l_2, \ldots, l_n \}, \text{ the set of lines}$$

$$E = \{ \{l_i, l_j\} : 1 \leqslant i, j \leqslant n \text{ and columns } l_i \text{ and } l_j \text{ of P intersect } \}$$

Thus $\{l_i, l_j\}$ is a (candidate) folding pair if and only if $\{l_i, l_j\}$ is not in E. An ordered folding set F transforms the (undirected) intersection graph into a Mixed Graph M(V,E,A), where A is the set of directed edges given by

$$A = \{ (l_i, l_j) : (l_i, l_j) \in F \}$$

Figure 2(a) shows the intersection graph of the PLA shown in Figure 1(a) and Figure 2(b) shows the (transformed) mixed graph due to an ordered folding set. We say that a folding set F includes a node v if there is another node u such that either (v,u) or (u,v) ϵ F.

In a mixed graph M(V,E,A), an Alternating Cycle is a sequence of edges $\langle (l_1, l_2), \{l_2, l_3\}, \ldots (l_{2k-1}, l_{2k}), \{l_{2k}, l_1\} \rangle$ such that the following conditions hold:

(i) $(l_{2i-1}, l_{2i}) \in A, \quad i = 1, 2, \ldots, k$

(ii) $\{l_{2i}, l_{2i+1}\} \in E, \quad i = 1, 2, \ldots, k-1$

(iii) $\{l_{2k}, l_1\} \in E.$

Informally, an alternating cycle is a cycle in which directed and undirected edges are traversed alternately. (A directed edge must be traversed only along the direction indicated by the edge). The following theorem proved in [HNS81] relates the realizability of an ordered folding set F and the alternating cycles in the mixed graph.

Theorem 1: Given an intersection graph $G(V,E)$ and an ordered folding set F, the set F is physically realizable if and only if the mixed graph $M(V,E,A)$ obtained from G and F does not contain an alternating cycle.

This theorem enables us to work with the intersection graph of a PLA instead of its Personality Matrix. In the remainder of this paper, we use intersection graphs to represent PLA's.

3. HEURISTICS FOR THE OPTIMAL FOLDING PROBLEM

The NP-Completeness of the optimal folding problem provides sufficient justification for us to consider heuristics for this problem. In this section, we analyze some variants of a generalized heuristic proposed in [HNS81] and obtain best possible bounds on their performance. The bounds indicate that these heuristics exhibit unbounded behavior in the worst case.

3.1 A General Performance Bound

A folding set[†] is maximal if no superset of F is also a folding set. An interesting class of heuristics for OFP consists of heuristics which produce maximal folding sets. It is easy to see that a bound on the ratio of the size of an optimal folding set to that of a maximal folding set is $\lfloor n/2 \rfloor$, where n is the number of lines. But as our next theorem shows, this bound is not achievable for $n>5$. We establish the theorem by first proving the following lemma.

Lemma 1: Let $G(V,E)$ be the n-node intersection graph of a PLA. Suppose $S = \{(v_1,v_2)\}$ is a one element maximal folding set for G. Then any folding set which includes either (or both) of v_1 and v_2 cannot contain more than 2 folding pairs.

Proof: If $n < 6$, there is nothing to prove, since the size of any folding set cannot exceed 2. So let $n \geqslant 6$ and suppose S_1 is a folding set of size $\geqslant 3$ which includes one of the nodes, say v_2. In S_1 let v_2 be folded with a node v_3 $(v_3 \neq v_1)$. This implies that the edge $\{v_2,v_3\}$ is not in E. Since $|S_1| \geqslant 3$, S_1 must contain at least one other folding pair (v_4,v_5), where v_4 and v_5 are distinct from v_1, v_2 and v_3. Note that in S, the folding pair (v_1,v_2) "prevented" the pair v_4,v_5 from being folded. It is easy to verify that this is possible only if G contains all of the following four edges: $\{v_1,v_4\}$, $\{v_1,v_5\}$, $\{v_2,v_4\}$ and $\{v_2,v_5\}$.

Since the edge $\{v_2,v_3\}$ is not present in E, it follows that folding v_3 with either v_4 or v_5 was prevented not by the folding pair (v_1,v_2) but because the edges $\{v_3,v_4\}$ and $\{v_3,v_5\}$ were present in G. Now however, since all the four edges $\{v_2,v_4\}$, $\{v_2,v_5\}$, $\{v_3,v_4\}$ and $\{v_3,v_5\}$ are present in E, it follows that folding v_2 with v_3 prevents the folding of v_4 with v_5. This is a contradiction, hence $|S_1| \leqslant 2$. □

[†] From now on, "folding set" means an ordered and realizable folding set.

Theorem 2: Let G(V,E) be an n-node intersection graph of a PLA P. Let OPT denote the size of an optimal folding set and MFS denote the size of a maximal folding set for G. Then,

$$\frac{OPT}{MFS} \begin{array}{lll} \leqslant & \lfloor n/2 \rfloor & \text{for} \quad 2 \leqslant n \leqslant 5 \\ \leqslant & \lfloor n/2 \rfloor - 1 & \text{for} \quad n > 5. \end{array}$$

This is the best possible result.

Proof: The bound $\lfloor n/2 \rfloor$ for $2 \leqslant n \leqslant 5$ is trivial. So let $n \geqslant 6$ and suppose that MFS = 1. Suppose the heuristic chooses the folding pair {(A,B)}. If OPT = $\lfloor n/2 \rfloor \geqslant 3$ (since $n \geqslant 6$), at least one of A and B must be included in any optimal folding set. This contradicts Lemma 1 since size of any such folding set is at most 2. Thus OPT $\leqslant \lfloor n/2 \rfloor - 1$.

We need not consider values of MFS $\geqslant$ 2, because the best possible upper bound would then only be 1/2 $\lfloor n/2 \rfloor$. This is weaker than the bound above.

An intersection graph which achieves the bound for n>5 is shown in Figure 3. The set $\{(v_1,v_2)\}$ is a one element maximal folding set, whereas an optimal folding set consisting of $|n/2|-1$ pairs is given by,

$\{(v_3,v_4),\ (v_5,v_6),\ \cdots\ ,\ (v_{n-3},v_{n-2}),\ (v_{n-1},v_n)\}$ if n is even, and

$\{(v_3,v_4),\ (v_5,v_6),\ \cdots,\ (v_{n-2},v_{n-1})\}$ if n is odd. □

3.2 A General Heuristic for OFP and its Variants

A framework for a class of heuristics which produce maximal folding sets has been presented (Algorithm MODEL) in [HNS81]. The steps of this heuristic are as follows. Let G(V,E) denote the intersection graph and let F denote the (initially empty) folding set produced by the heuristic. The initial value of the set V_t is V.

(1) If V is empty, output F and stop. Otherwise, select a node v from V.

(2) Let U = {u: u ϵ V_t, u$\neq$v and (u,v) not in E}.

(3) If U is empty, go to step (5). Otherwise, select a node u ϵ U.

(4) If (v,u) when added to F causes an alternating cycle, delete u from U and go to step (3). Otherwise, add (v,u) to F, delete v and u from V and V_t and go to step (1).

(5) Delete v from V and go to step (1).

We can derive a number of variants from this general framework by specifying how nodes v and u should be selected in steps (1) and (3). A selection criterion suggested in [HNS81] is based on the degrees of the nodes of G. This reference suggests and presents an intuitive discussion of the variants (b) through (e) listed below[†]:

(a) *RANDOM* : Choose v and u randomly.

(b) *MAX-MAX* or MAX^2 : Choose v and u with the largest degree from the sets V and U respectively.

(c) *MIN-MIN* or MIN^2 : Choose v and u with the smallest degree from V and U respectively.

(d) *MIN-MAX* : Choose v to be a node of least degree and u to be a node with the largest degree from V and U respectively. This is the variant chosen by the authors of [HNS81] for implementation.

(e) *MAX-MIN* : Choose v of largest degree and u of smallest degree from V and U respectively.

In all cases, ties are resolved arbitrarily. In what follows, we present the worst case bounds for the above variants (using the results of the previous section). Here, degree(v) denotes the degree of a node v in the (undirected) intersection graph.

Corollary 1: For the *RANDOM* heuristic,

$$\frac{OPT}{RANDOM} \begin{array}{ll} \leqslant \lfloor n/2 \rfloor & \text{for} \quad 2 \leqslant n \leqslant 5 \\ \leqslant \lfloor n/2 \rfloor - 1 & \text{for} \quad n > 5. \end{array}$$

This is the best possible result.

Proof: Follows directly from the proof of Theorem 2.□

[†] The terminology is ours.

Corollary 2: For the MAX^2 and $MAX-MIN$ heuristics (HEU),

$$\frac{OPT}{HEU} = 1 \quad \text{for} \quad 2 \leqslant n \leqslant 5$$
$$\leqslant \lfloor n/2 \rfloor - 1 \quad \text{for} \quad n > 5.$$

This is the best possible result.

Proof (sketch): In view of Theorem 2, it is only necessary to show that MAX^2 and $MAX-MIN$ are optimal for all graphs with at most 5 nodes. For n=2 and n=3, there is nothing to prove. We present the proof for n=4 that if MAX^2 produces a folding set of size 1, the optimal folding set is of size 1. The proof for n=5 is similar.

Let V = {A,B,C,D} and let (A,B) be the folding pair chosen by MAX^2. Assume that A has the largest degree among the four nodes. Suppose that OPT = 2 and without loss of generality (as will become apparent), let {(A,C), (B,D)} denote an optimal folding set. Thus E does not contain the edges {A,B}, {A,C}, and {B,D}. Hence degree(A) $\leqslant$ 1. Since {(A,B)} is a maximal folding set and the edge {A,C} is absent in E, it follows that {C,D} is in E. Now, if the edge {A,D} is in E, degree(D) $\geqslant$ 2, while if {A,D} is not in E, degree(A) = 0 and degree(D) = 1. In both the cases, we contradict the maximality of degree(A). Hence OPT =1 in this case. □

The bound given by Theorem 2 is not tight for the two heuristics MIN^2 and $MIN-MAX$, which first select a node with the least degree. The reason for this is that when one folding pair (such as (v_1, v_2) of figure 3) prevents a large number of other folding pairs, one of the nodes in the "offending pair" no longer has the smallest degree and so will not get chosen by MIN^2 or $MIN-MAX$. The following theorem shows the achievable bound.

Theorem 3: Let G(V,E) be the n-node intersection graph of a PLA. Let OPT denote the size of an optimal folding set for G and let HEU denote the size of a folding set produced by MIN^2 or $MIN-MAX$. Then for n>6, $\frac{OPT}{HEU} \leqslant \lfloor (n-1)/3 \rfloor$ and this is the best possible result.

Proof: Consider an intersection graph G for which MIN^2 (or $MIN-MAX$) produces a folding set of size 1. In what follows, we obtain a bound on the size of an optimal folding set for G.

Let (A,B) denote the folding set produced by MIN^2. As before, let A be the node in G of least degree. Let OS = $\{(p_1,q_1), (p_2,q_2), \cdots, (p_r,q_r)\}$ be an optimal folding set of size r for G. If OS includes either (or both) of A and B, then r $\leqslant$ 2 by Lemma 1. Thus assume that OS does not contain either A or B. Let P = $\{p_1, p_2, \cdots, p_r\}$ and let Q = $\{q_1, q_2, \cdots, q_r\}$. We have the following claim:

Claim: There is a node q_i in Q which is not connected by an undirected edge to any node in P.

Proof of Claim (sketch): Suppose not. That is, assume that every node q_i in Q is connected to at least one node in P. Then, in the mixed graph resulting from OS, we can find an arbitrarily long sequence of directed and undirected edges by starting at p_1 and moving to and fro between sets P and Q via directed and undirected edges respectively. However, since there are only r distinct directed edges, any such sequence containing 2r or more elements must contain an alternating cycle, which implies that OS is unrealizable. Hence the claim. □

To complete the proof of Theorem 3, note that since (A,B) prevented the folding of all pairs in OS, A and B must be connected to all nodes in P∪Q. Thus degree(A) $\geqslant 2r$. From the above claim, there is a node q_i in Q such that degree(q_i) $\leqslant n-r-1$. Since degree(A) $\leqslant$ degree(q_i), it immediately follows that $r \leqslant \lfloor (n-1)/3 \rfloor$.

As in Theorem 2, the best possible bound when MIN^2 produces folding sets of size 2 or more is still 1/2 $\lfloor n/2 \rfloor$ and this is weaker than the above bound.

Our proof also shows how we can exhibit an intersection graph which achieves the bound. This is shown in Figure 4. We simply choose P and Q to be cliques of size $\lfloor (n-1)/3 \rfloor$. □

We now state the exact bounds for MIN^2 and $MIN\text{-}MAX$ for all values of n. The proofs for the range n<6 are similar to the proof presented in Corollary 2.

Corollary 3: For the heuristic MIN^2,

$$\begin{aligned} \frac{OPT}{MIN^2} &\leqslant \lfloor n/2 \rfloor \quad \text{for} \quad 2 \leqslant n \leqslant 5 \\ &\leqslant 2 \quad \text{for} \quad n = 6, \quad \text{and} \\ &\leqslant \lfloor (n-1)/3 \rfloor \quad \text{for} \quad n > 6. \end{aligned}$$

This is the best possible result.

Corollary 4: For the heuristic $MIN\text{-}MAX$,

$$\begin{aligned} \frac{OPT}{MIN\text{-}MAX} &= 1 \quad \text{for} \quad 2 \leqslant n \leqslant 5 \\ &\leqslant 3/2 \quad \text{for} \quad n = 6, \quad \text{and} \\ &\leqslant \lfloor (n-1)/3 \rfloor \quad \text{for} \quad n > 6. \end{aligned}$$

This is the best possible result.

4. BOUNDS FOR SPECIAL CLASSES OF PLA'S

The results presented above indicate that heuristics based on degrees do not perform well in the worst case. It is an open question whether there is a heuristic which achieves a constant worst case ratio for any intersection graph (PLA). For some special classes of graphs however, a constant ratio can be achieved using the following heuristic: obtain an independent set and convert it into a folding set by pairing up the nodes arbitrarily. In general, this heuristic does not produce maximal folding sets. However, this heuristic leads to a bound of 2 for bipartite graphs. Moreover, the greedy heuristic for finding independent sets presented in [PY81] makes it possible to extend this approach to (a) Planar graphs and (b) Graphs whose degrees are bounded by a constant K. The greedy heuristic consists of repeatedly selecting a node v of least degree and discarding the nodes adjacent to v. This heuristic produces an independent set of size at least n/6 for planar graphs (since every planar graph has a node of degree $\leqslant 5$; see [PY81] for a proof of the tight lower bound 4n/21) and at least n/(K+1) for constant degree bounded graphs. Since in each case the optimal folding set is of size at most n/2 we have the following:

Lemma 2: There is a polynomial time heuristic for

(a) Planar graphs, with $\frac{OPT}{HEU} \leqslant 6$, and for

(b) Graphs with a constant bound (K) on node degree, with $\frac{OPT}{HEU} \leqslant K+1$.

For planar graphs, as mentioned above, the bound is not tight. A bound of 4 is achievable (let n = 8d; consider 2d cliques, each containing 4 nodes). For (b), the bound is the best possible (let n = 2d(K+1); consider 2d cliques, each containing K+1 nodes).

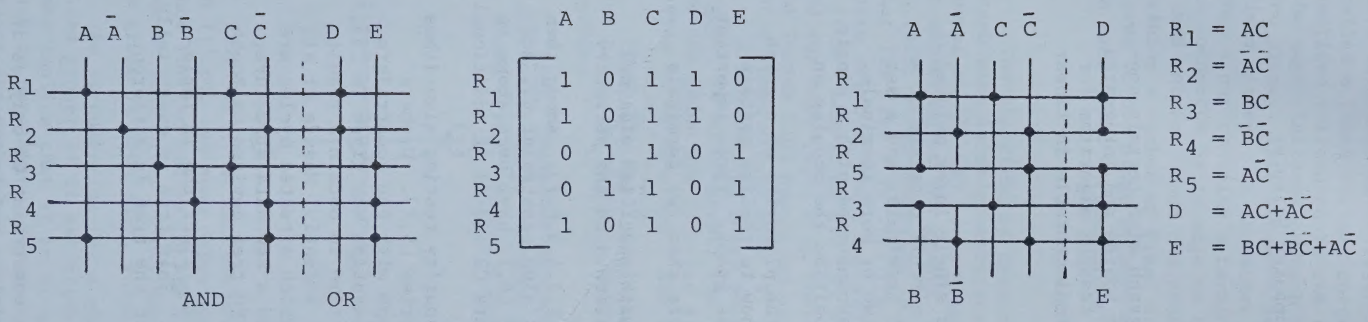
REFERENCES

[HNS81] Hachtel, G.D., Newton, A.R., and Sangiovanni-Vincentelli, A.L., "An Algorithm for Optimal PLA Folding", IBM Research Report RC 8668, January 1981.

[LVVS82] Luby, M., Vazirani, V., Vazirani, U., and Sangiovanni-Vincentelli, A.L., Private Communication.

[MC80] Mead, C.A., and Conway, L., "Introduction to VLSI Systems", Addison-Wesley, 1980.

[PY81] Papadimitriou, C.H., and Yannakakis, M., "Worst-Case Ratios for Planar Graphs and the method of Induction on Faces", Proc. of the 22nd IEEE Symp. on FOCS, October 1981, pp 358-363.


A
$\bar{A}$
B
$\bar{B}$
C
$\bar{C}$
D
E
R_1
R_2
R_3
R_4
R_5
AND
OR
A B C D E
R_1 1 0 1 1 0
R_2 1 0 1 1 0
R_3 0 1 1 0 1
R_4 0 1 1 0 1
R_5 1 0 1 0 1
A
$\bar{A}$
C
$\bar{C}$
D
R_1
R_2
R_5
R_3
R_4
B
$\bar{B}$
E
$R_1 = AC$
$R_2 = \bar{A}\bar{C}$
$R_3 = BC$
$R_4 = \bar{B}\bar{C}$
$R_5 = A\bar{C}$
$D = AC+\bar{A}\bar{C}$
$E = BC+\bar{B}\bar{C}+A\bar{C}$

(a) (b) (c)

Figure 1: A PLA, its Personality Matrix and its Folded Version

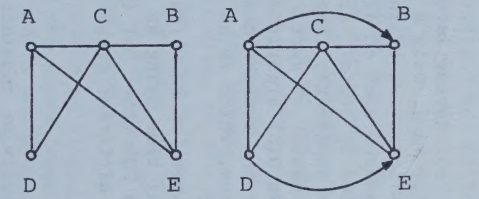

A
C
B
D
E
A
C
B
D
E

Figure 2: Intersection Graph and Mixed Graph due to $F = \{(A,B),(D,E)\}$

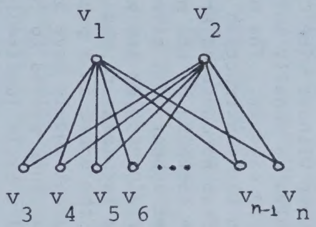

v_1
v_2
v_3
v_4
v_5
v_6
...
v_{n-1}
v_n

Figure 3

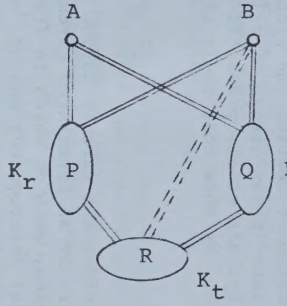

A
B
P
Q
R
K_r
K_r
K_t

Figure 4

$r = (n-1)/3$

$t = n-2r-2$

K_m: Clique of m nodes

═ and ═ ═ ═ denote edges between all pairs of nodes.

═ ═ ═ :Included for MIN-MAX only.

ON AN EFFICIENT IMPLEMENTATION OF A PLANARITY TESTING ALGORITHM FOR A GRAPH WITH LOCAL CONSTRAINTS

KAZUO NAKAJIMA and MOSES SUN
Department of Electrical Engineering and Computer Science
Texas Tech University
Lubbock, Texas 79409

ABSTRACT

The problem of testing the planarity of a graph with certain constraints arises in the design of printed circuit boards and integrated circuits. Uyehara et al. presented a planarity testing algorithm for such a graph. We show that their algorithm is implementable in linear time.

1. INTRODUCTION

We consider the circuit layout problem on a single layer printed circuit board or an integrated circuit chip. A circuit consists of a set of modules, each of which is itself a set of terminals, and a set of electrical interconnections or "nets" among two or more terminals. Each net specifies a set of terminals to be interconnected with a single conducting path or "wire". The problem is to position the modules on the board or chip without wire crossings.

This problem is usually solved by testing the planarity of a graph representing the circuit. The main problem is how to realize certain constraints in the construction of the respective graph. Three important constraints often encountered are:

- (C1) The contacts of the modules having more than two terminals must appear in a given sequence.
- (C2) Certain modules are placed on board with specified side up.
- (C3) The terminals of the circuit are positioned on the periphery of the board in a prescribed sequence.

Various graph models have been proposed [1,2,3,4,5,6,7]. Among them a wheel [2] has been proved to be useful in realizing constraint C1, and a star [5] and a cycle (undirected and directed) [6,7] have been shown to be most powerful in the realization of constraints C2 and C3 if additional information is kept appropriately (see Fig. 1).

On the other hand, only a few practical planarity testing algorithms for graphs with these constraints have been reported [5,7]. Uyehara et al. [5] presented such an algorithm for a graph with the constraints being represented by stars. Although no implementation was given in [5], their algorithm apparently takes at least $O(n^2)$ time to obtain a result, where n is the number of vertices in the graph. Recently, Masuda et al. [7] developed an $O(n)$ algorithm for a graph in which directed cycles are used to realize the constraints. They introduced a new data structure, called a PQR tree, which is an extention of the PQ tree devised by Booth and Lueker [8].

In this paper, we show that the Uyehara et al. algorithm is implementable in $O(n)$ time using an extended PQ tree. This tree is essentially the same as the PQR tree, but the manipulation of the tree is different because of the different graph model.

2. THE GRAPH MODEL

For simplicity we consider a circuit having constraint C2 only as in the Uyehara et al. paper [5]. In their graph model, a module with two or more terminals is represented by a star where the vertex in the center, called a fixed vertex, corresponds to the module and the surrounding vertices correspond to the terminals. Then the constraint C2 implies that the edges incident on each fixed vertex must appear in a fixed sequence in the plane. This sequence is specified as the local constraint for each fixed vertex. On the other hand, each net is represented by a vertex. When a net contains a terminal of a module, the corresponding net vertex is connected by an edge to the terminal vertex which is adjacent to the fixed vertex. Then the problem is stated as follows: Given a graph containing a number of fixed vertices, determine whether the graph can be drawn in the plane without crossing of edges such that the edges incident on each fixed vertex appear in a prescribed order.

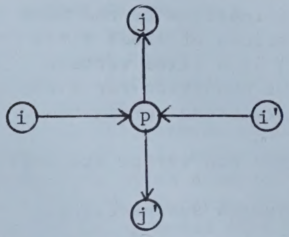
3. THE ALGORITHM

The algorithm of Uyehara et al. [5] for testing the planarity of a graph with local constraints is an extention of the algorithm due to Lempel et al. [9] for a graph without constraints, which is based on the st-numbering of the graph. Given a biconnected graph $G(V,E)$ and an edge $(s,t)\varepsilon E$, an st-numbering is a 1-1 function $f : V \rightarrow \{1,2,\ldots,|V|\}$ such that (i) $f(s) = 1$, (ii) $f(t) = |V|$, and (iii) for every $v \varepsilon V - \{s,t\}$, there exist at least two adjacent vertices u and w such that $f(u) < f(v) < f(w)$. Lempel et al. [9] proved that for any biconnected graph $G(V,E)$ and any edge $(s,t) \varepsilon E$, there exists an st-numbering. Furthermore, Even and Tarjan [10] developed an $O(n+m)$ algorithm for computing an st-numbering for a graph $G(V,E)$ with $|V| = n$ and $|E| = m$.

A biconnected graph whose vertices are st-numbered is called an st-graph. In an st-graph, the vertices are identified by their st-numbers and the edges are directed from low to high. We now consider a fixed vertex in an st-graph. The following lemma is easily obtained.

Lemma 1: For any planar st-graph with fixed vertices, all lower (and hence all higher) vertices adjacent to each fixed vertex appear together.

Proof: Suppose, on the contrary, that lower vertices i, i' and higher vertices j, j' which are adjacent to vertex p are intermingled as shown below. From the property of the st-numbering, there exist a directed path from vertex 1 to vertex i and a directed path from vertex 1 to vertex i' such that none of the paths include any vertex higher than or equal to vertex p. Let i" be the last vertex which is common on these paths. Then the edges on the path from i" to i and those on the path from i" to i' together with the edges (i,p) and (i',p) constitute a cycle C_i, if the directions of the edges are ignored.



Similarly, we can obtain a cycle C_j consisting of the edges (j,p) and (j',p), and some other edges whose endpoints are greater than p. Since the two cycles C_i and C_j meet at vertex (or point) p, they must meet at some other point, due to the Jordan curve theorem. Therefore, the graph is nonplanar, which is a contradiction. Q.E.D.

An st-numbering for a graph (or interchangeably an st-graph) is called proper if it satisfies the condition of the above lemma. The lemma implies that if an st-numbering for a graph is not proper, then the graph is nonplanar. Furthermore, in a proper st-graph the local constraint for each fixed vertex can be represented by a pair of conditions:

(i) the input condition which is the sequence of all lower vertices adjacent to the fixed vertex, and (ii) the output condition which is the sequence of all higher vertices adjacent to the fixed vertex.

Following Even [11], we now define a bush form. Let $G(V,E)$ be an st-graph, where $V = \{1,2,\ldots,n\}$. Let $V_k = \{1,2,\ldots,k\}$ and $G_k(V_k,E_k)$ be the directed graph induced by V_k; that is, E_k consists of all edges of $G(V,E)$ whose endpoints are both in V_k. Then the directed graph B_k is defined as follows. G_k is a subgraph of B_k, and B_k contains all edges of G_k which emanate from vertices of V_k and enter in G, vertices of $V - V_k$. These edges are called virtual edges and the vertices they enter in B_k are called virtual vertices. These virtual vertices are labelled as their counterparts in G, but they are kept separately. Therefore, several virtual vertices may have the same label (i.e., st-number), but each of them has exactly one entering edge. Since $(1,n)$ is always an edge in G, we can draw B_k in the following form: Vertex 1 is drawn at the top level and all virtual vertices appear on one horizontal line. The remaining vertices of G_k are drawn in such a way that vertices with higher labels are drawn lower. Such a realization is called a bush form.

Let v be a separation vertex of B_k. Then this vertex v determines a set of connected components each including v. And v is the lowest vertex in each of the components, exept the one which contains vertex 1 (in case $v > 1$). Each of these components is a sub-bush; that is, it has the same structure as a bush form, exept its lowest vertex is v rather than 1. These sub-bushes can be permuted around v in any one of all possible permutations. In addition, each of the sub-bushes can be flipped over. These transformations maintain the bush form, and they are called equivalence transformations. Note that when a fixed vertex is involved directly or indirectly in an equivalence transformation, not all of the permutations and flippings are allowed.

We now describe the Uyehara et at. algorithm for testing the planarity for a graph containing fixed vertices.

```
Algorithm: (Assume a proper st-graph is given.)
  Step 1:  $k \leftarrow 1$.
  Step 2:  Draw $B_k$ of G.  (If k is a fixed vertex, its leaving
           edges are placed in the order specified by the output
           condition for k.)
  Step 3:  Apply necessary equivalence transformations to the bush
           form of $B_k$ so that all virtual vertices of label $k + 1$
           appear consecutively and if $k + 1$ is a fixed vertex,
           in the order specified by the input condition for $k + 1$.
           If this cannot be done, then the algorithm terminates
           unsuccessfully.
  Step 4:  Join all virtual vertices $k + 1$ into one vertex and pull
           it up to the horizontal line.
  Step 5:  If $k = n$, then the algorithm terminates successfully.
  Step 6:  $k \leftarrow k + 1$.
  Step 7:  Go to Step 2.
```

An example of an st-graph is given in Fig. 2. The generation processes of bush forms for the graph in Fig. 2 are shown in Fig. 3, using the above algorithm. Following Lempel et al. [9], Uyehara et al. [5] used a formula to represent a bush form. A product and a concatenate were used to realize permutaions and flippings, respectively as in [9]. Furthermore, a star '*' was introduced in the realization of the effects of a fixed vertex. The time complexity of the algorithm based on the formula manipulation is at least $O(n^2)$.

4. EXTENDED PQ-TREES

4-1. PQ-trees: We briefly explain PQ-trees invented by Booth and Lueker [8]. A PQ-tree is a rooted ordered tree with three types of vertices, P-nodes, Q-nodes, and leaves. In our application, a P-node represents a separation vertex v of B_k, and a Q-node represents a maximal biconnected component of B_k. And the leaves represent the virtual edges, or equivalently the virtual vertices.

The frontier of a PQ-tree T, denoted by F(T), is obtained by reading its leaves from left to right. Two trees T and T' are said to be equivalent, written as $T \equiv T'$, if one may be obtained from the other by applying zero or more equivalence transformations: (i) arbitrarily permuting the children of a P-node, and (ii) reversing the children of a Q-node. Let $CON(T) = \{F(T') | T' \equiv T\}$.

Given a PQ-tree T and a set S of the leaves labeled k + 1, the procedure to make a new PQ-tree T' such that

$CON(T') = \{\pi \mid \pi \in CON(T)$ and the elements of S appear together in $\pi\}$

is called the reduction with respect to S. The new tree is denoted by REDUCED(T,S). The pertinent subtree of T with respect to S is the subtree of minimum hight whose frontier contains all elements of S. The root of the pertinent subtree is denoted by ROOT(T,S). The pruned pertinent subtree of T with respect to S is the smallest connected subgraph which contains all of the leaves in S. This subtree is denoted by PRUNED(T,S).

Booth and Lueker [8] obtained an algorithm for producing a new PQ-tree REDUCED(T,S) from a given PQ-tree T and a given set S of leaves, which runs in $O(|PRUNED(T,S)|)$ time. Using this algorithm, they presented an $O(|V|)$ algorithm for testing the planarity of a graph G(V,E) without constraints.

4-2. Extended PQ-trees: The Booth-Lueker planarity testing algorithm is an implementation of the Lempel et al. algorithm using PQ-trees. The Uyehara et al. algorithm is an extention of the Lempel et al. algorithm. Therefore, it is natural to obtain an appropriate extention of PQ-trees to implement the Uyehara et al. algorithm efficiently.

As is seen from the statements of the algorithm described in the previous section, when a fixed vertex is selected for vertex k or k + 1, the lower vertices adjacent to the fixed vertex must always appear together in the order specified by its input condition, and the higher vertices adjacent to it must appear together in the order specified by its output condition. Therefore, in order to realize these fixed sequences of the adjacent vertices, a new type of vertex must be introduced to a PQ-tree. This is an S-node which plays the same role as the star does in the formula representation of bush forms proposed by Uyehara et al. [5]. In other words, the children of an S-node are fixed in order. A PQ-tree which also contains S-nodes is called an extended PQ-tree or a PQS-tree. Because of the addition of S-nodes to PQ-trees, the manipulation of extended PQ-trees becomes more complex than that of regular PQ-trees. However, the basic principles of PQ-tree algorithms can be easily carried over to algorithms using extended PQ-trees. Therefore, in this paper we will briefly explain these principles without going into the details of extended PQ-trees and their associated algorithms.

The main procedure used for an efficient implementation of Uyehara et al. algorithm is the procedure called X-REDUCE(T,S) which is similar to the procedure to produce a new PQ-tree REDUCED(T,S). This procedure scans a PQS-tree from children to parents, starting with the leaves labeled k + 1 in S. Each time a new node is scanned, it modifies the current PQS-tree so that the descendants of the node which have label k + 1 appear together and if necessary, in the specified order. This is done through a technique of *template matching*. For each node and its children, their pattern is matched to one of the templates' *patterns*. If one such template's pattern is found, the original subtree rooted at the node is replaced within the tree by the template's *replacement* which is a new subtree rooted at the same node. And a next node is scanned. This process is continued until ROOT(T,S), the root of the pertinent subtree of T with respect to S, is scanned. However, if no matching occurs at certain stage, the procedure concludes that the original PQS-tree cannot be reduced for the specified set S, and puts out the null tree, denoted by $T(\phi,\phi)$, which has no node at all. In both cases, the new tree created by procedure X-REDUCE(T,S) is denoted by X-REDUCED(T,S). Note that it is not difficult to obtain appropriate templates for X-REDUCE(T,S). The following lemma is obtained along the same line as the one for the procedure to produce REDUCED(T,S) given in [8].

Lemma 2: Procedure X-REDUCE(T,S) requires $O(|\text{PRUNED}(T,S)|)$ time.

5. AN EFFICIENT IMPLEMENTATION

Using procedure X-REDUCE(T,S), we can implement the Uyehara et al. algorithm efficiently. Before we proceed, a few more definitions are necessary. A node is said to be *full* if all of its descendants are in S. The *universal* tree is a PQ-tree which has a single P-node for its root and a leaf for every element of U. It is denoted by T(U,U).

A planarity testing algorithm for a graph with local constraints using extended PQ-trees is presented below.

```
procedure X-PLANAR(V,E):
  begin
    if $|E| > 3|E| - 6$ then return false;
    compute an st-numbering for V;
    if the st-numbering is not proper  then return false;
    U := $\{(1,i) | (1,i) \in E\}$;
    T := T(U,U);
    for j := 2 to $|V| - 1$ do
      begin
        S := $\{(i,j) | (i,j) \in E, i < j\}$;
        T := X-REDUCED(T,S);
        if T = $T(\phi,\phi)$  then return false;
        S' := $\{(j,k) | (j,k) \in E, j < k\}$;
        if j is a fixed vertex
          then replace the full children of ROOT(T,S) and their
                 descendants by the elements of S' in the order
                 specified by the output condition of j
          else if ROOT(T,S) is either a Q-node or an S-node
                 then replace the full children of ROOT(T,S) and
                        their descendants by T(S',S')
                 else replace ROOT(T,S) and its descendants by
                        T(S',S');
        U := U - S   S'
      end;
    return true
  end;
```

The following theorem is obtained in a similar way to the one for procedure PLANAR(V,E) given in [8].

Theorem 1: Procedure X-PLANAR(V,E) is correct and requires at most $O(|V|)$ time.

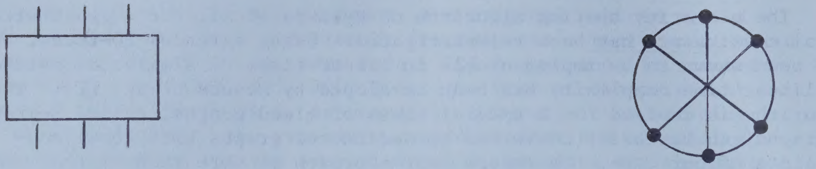
An illustrative process of the procedure is shown in Fig. 4.

6. CONCLUSIONS

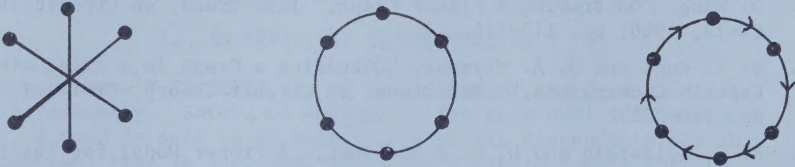
The planarity testing algorithm of Uyehara et al. for a graph with local constraints has been reinvestigated. Using extended PQ-trees, it has been shown to be implementable in linear time. A similar algorithm of linear time complexity has been developed by Masuda et al. [7]. Their algorithm is devised for a special class of mixed graphs, called L-graphs. L-graphs can be easily converted to undirected graphs with local constraints of our type. Therefore, our approach is more general than theirs. Nevertheless, it is interesting to compare the performances of both algorithms by actually implementing them. The details of the implementation of our algorithm is found in [12].

REFERENCES

1. O. Wing, "On Drawing a Planar Graph," IEEE Trans. on Circuit Theory, CT-13, 1966, pp. 112-114.
2. W. L. Engl and D. A. Mlynski, "Embedding a Graph in a Plane with Certain Constraints," IEEE Trans. on Circuit Theory, CT-17, 1970, pp. 250-252.
3. A. J. Goldstein and D. G. Schweikert, "A Proper Model for Testing the Planarity of Electrical Circuits," Bell System Tech. J., 52, 1973, pp. 135-142.
4. M. C. Vanlier and R. H. Otten, "On the Mathematical Formulation of the Wiring Problem," Int. J. of Circuit Theory and Applications, 1, 1973, pp. 137-147.
5. T. Uyehara, H. Shiraishi, O. Takahashi, and T. Kojima, "Embedding a Graph in a Plane with Local Constraints," IEEE Int. Symp. on Circuits and Systems, San Francisco, 1974, pp. 22-25.
6. W. M. vanCleemput, "Mathematical Models for the Circuit Layout Problem," IEEE Trans. on Circuits and Systems, CAS-23, 1976, pp. 759-767.
7. S. Masuda, T. Kashiwabara, and T. Fujisawa, "A Layout Problem on Single Layer Printed Circuit Board," IECE of Japan, Tech. Rep. CAS81-19, 1981, pp. 93-100.
8. K. S. Booth anf G. S. Lueker, "Testing for the Consecutive Ones Property, Interval Graphs, and Graph Planarity Using PQ-tree Algorithms," J. of Comp. and Sys. Sciences, 13, 1976, pp. 335-379.
9. A. Lempel, S. Even, and I. Cederbaum, "An Algorithm for Planarity Testing of Graphs," Theory of Graphs: Int. Symp., Rome, 1966, P. Rosenstiehl, Ed., Gordon and Breach, New York, 1967, pp. 215-232.
10. S. Even and R. E. Tarjan, "Computing an st-numbering," Theoretical Comp. Sci., 2, 1976, pp. 339-344.
11. S. Even, Graph Algorithms, Computer Science Press, Potomac, MD, 1979, Chap. 7.
12. K. Nakajima and M. Sun, "Efficient Planarity Testing for Partially Oriented Graphs," in preparation.



(a) Module (b) Wheel



(c) Star (d) Cycle (e) Directed cycle

Fig. 1. Graph models for a muti-terminal module.

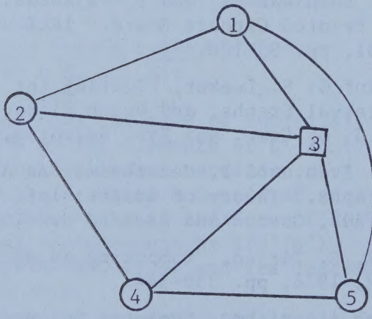


Fig. 2. An st-graph.

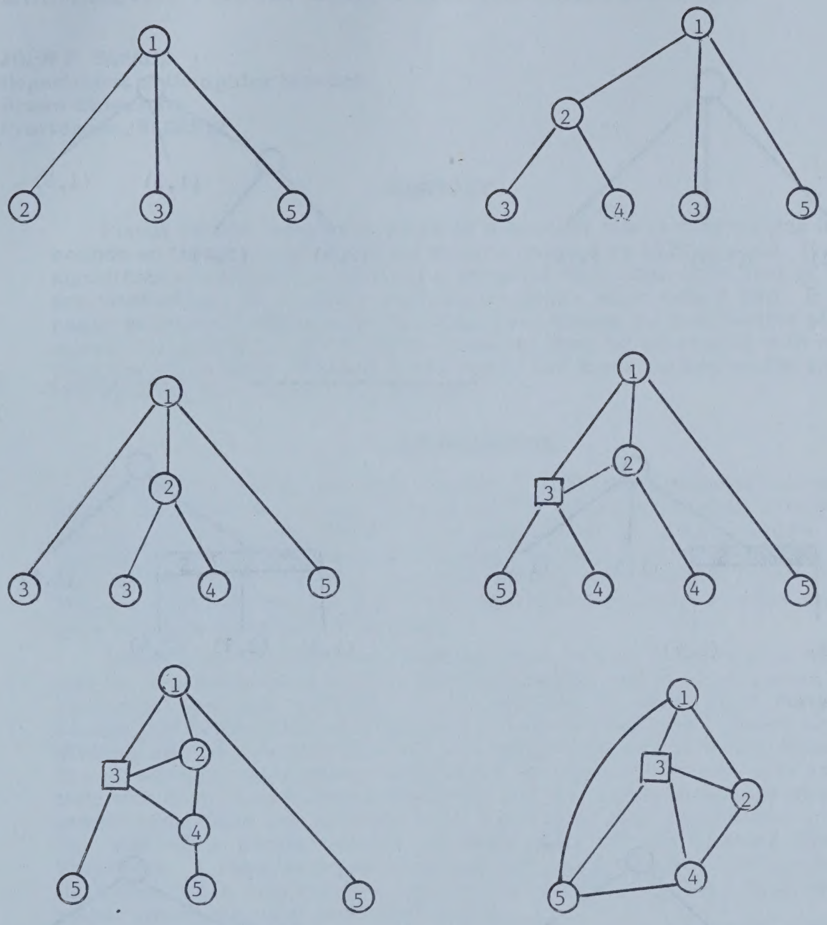

1
2
3
5
1
2
3
4
3
5
1
2
3
3
4
5
1
2
3
5
4
4
5
1
3
2
4
5
5
5
1
3
2
4
5

Fig. 3. Generation processes of bush forms for the graph in Fig. 2.
Vertex 3 is assumed to be a fixed vertex with input condition = (1,2) and output condition = (5,4).

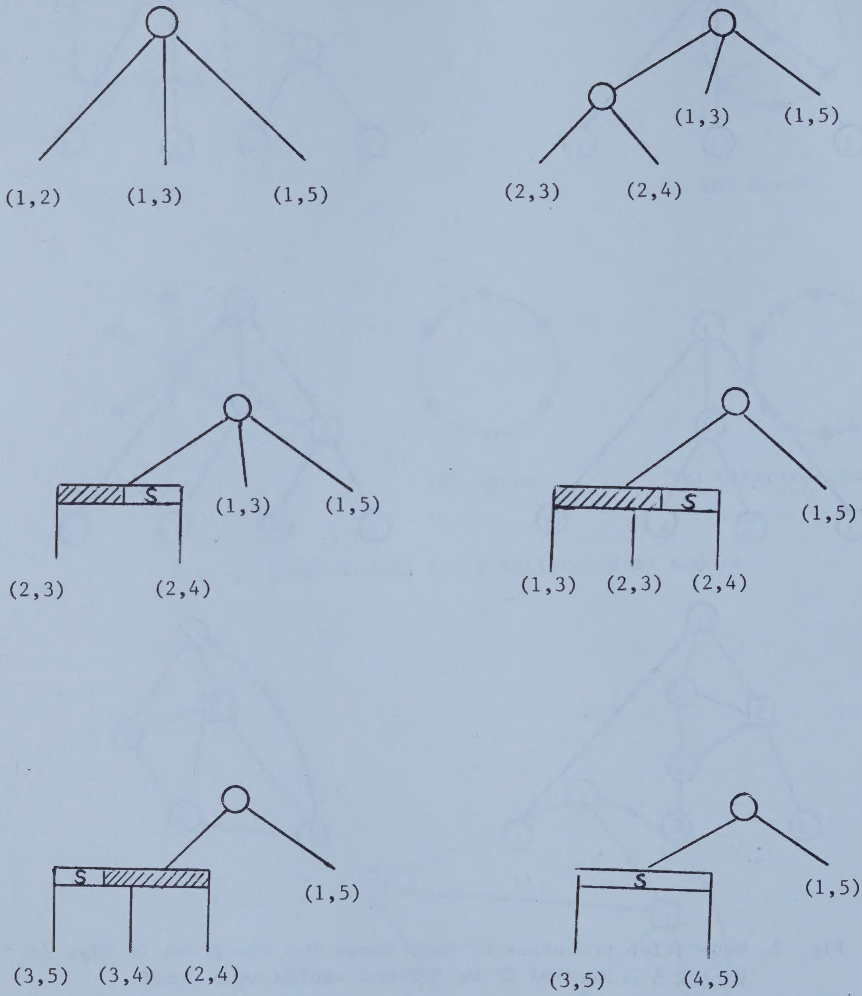

(1,2)
(1,3)
(1,5)
(1,3)
(1,5)
(2,3)
(2,4)
S
(1,3)
(1,5)
(2,3)
(2,4)
S
(1,5)
(1,3)
(2,3)
(2,4)
S
(1,5)
(3,5)
(3,4)
(2,4)
S
(1,5)
(3,5)
(4,5)

Fig. 4. Illustrative processes of X-PLANAR(V,E) applied to the graph in Fig. 2.

MULTILECTIVE PLANAR CIRCUIT SIZE (Extended Abstract)†

JOHN E. SAVAGE
Department of Computer Science
Brown University
Providence, RI 02912

ABSTRACT

Planar circuit complexity plays an important role in determining lower bounds on the performance of algorithms realized by VLSI circuits. If such algorithms are allowed to read input variables more than once, that is, they are *multilective*, then planar circuit complexity must reflect this. In this paper we present methods for deriving lower bounds on multilective planar circuit complexity in which inputs variables may be associated with more than one input node. We also derive upper and lower bounds on the size of this measure for a variety of problems.

1. INTRODUCTION

The planar circuit size of a function f, $C_p(f)$ is the minimal number of circuit elements in a planar circuit for f in which each input variable is assigned a unique input node. It is measure of the area required by a memoryless VLSI circuit for f, and it also provides a lower bound on the product AT^2 for a VLSI chip that computes f in chip area A and in T cycles [10]. These results hold for chip algorithms that read each input variable exactly once in a data-independent manner.

Lipton and Tarjan [6] have derived lower bounds to the planar circuit size for superconcentrators, the shifting function, and Boolean matrix multiplication that are quadratic in the number of their input variables. Savage [10] has derived such bounds for transitive functions, binary integer division, and certain normalized powers and reciprocals of binary numbers. In each of these cases, planar circuits are assumed to have one node associated with each input variable. Such circuits are said to be *semelective*. To permit more than one instance of an input node may significantly reduce the size of a planar circuit, as illustrated by the Discrete Fourier Transform. It requires a planar circuit size quadratic in n, the number of its inputs, when realized by a semelective circuit, but has a *multilective* planar circuit size of at most $O(n^{3/2}\log n)$.

Multilective planar circuit size provides a lower bound to the measure AT^2 for chip algorithms that read inputs more than once at more than one place in a data-independent manner. This result is established in a manner analogous to that given in [10] for algorithms that read inputs at at most one place on a chip. This relationship permits a generalization of the results of Kedem and Zorat [5] for multilective VLSI chip algorithms who derive lower bounds to this product measure for transitive functions, and those of Yao [15] for certain arithmetic functions.

In this paper we derive lower bounds on the multilective planar circuit size of functions in terms of their flow properties, and on predicates in terms of the degree to which their inputs can be separated. The bounds are

---

This research was supported in part by the National Science Foundation under Grant ECS-80-24637.

stated in terms of the number of times that input variables are read. If f is a function on n inputs, and in a planar circuit for f there are μn instances of nodes associated with the inputs, then for many functions, such as the Discrete Fourier Transform, the smallest planar circuit for f must have size $\Omega(n^2/\mu^2)$. Since it must also have size at least μn, this implies that its size is at least $\Omega(n^{4/3})$. This should be compared with the upper bound stated above. We derive such bounds for a variety of functions and predicates, and illustrate them with upper bounds for a few problems.

2. PLANAR CIRCUIT SIZE

A *planar circuit* is defined to be a labeled, directed, acyclic graph which is planar. Nodes of in-degree zero are *source nodes* (or inputs) labeled with the names of variables. Internal node labels are drawn from a set of functions called a *basis*. Circuits for which the variables take values over $\{0,1\}$ and for which the basis elements are functions over this set are Boolean. We do not limit our attention to Boolean circuits and functions.

Definition 1: The *circuit complexity* of a function $f:X^n \to X^m$, denoted C(f), is the size (number of logic elements) in the smallest logic circuit for f relative to a *basis* Ω, which is taken for simplicity to be the set of all functions $h:X^2 \to X$. The *planar circuit complexity* $C_p(f)$ of a function f is the size of the smallest planar circuit for f over this same basis. If a planar circuit has μn nodes associated with inputs of a function of n input variables, $\mu \geq 1$, then it is said to be *multilective of order* μ, and the measure is denoted $C_p^{(\mu)}(f)$.

It is straightforward to show that for all functions $f:\{0,1\}^n \to \{0,1\}^m$

$$C(f) \leq C_p(f) \leq C_p^{(\mu)}(f) \leq C_p^{(1)}(f) \leq 6[C(f)]^2$$

for $\mu \geq 1$. (See for example [10].) Thus, planar circuit size is never more than quadratic in standard circuit size. Also, it is known [8] that for any $0 < \delta < 1$ a fraction of at least $1 - 2^{-\delta m 2^n}$ of the functions $f:\{0,1\}^n \to \{0,1\}^m$ have

$$C(f) \geq m(2^n/n)(1-\delta-o(n))$$

if $\log m = o(n)$, and a simple construction [10] demonstrates that

$$C_p^{(1)}(f) \leq 7m2^n$$

for all such functions and all values of m and n. Thus, most functions on n variables have standard and planar circuit sizes that differ by at most a slowly growing factor in n.

Lipton and Tarjan use the Planar Separator Theorem (PST) [6] to derive lower bounds to the semelective planar circuit size of certain Boolean functions. The PST states that given any non-negative cost function on the vertices of a planar graph of N vertices, by removal of a set C of at most $O(\sqrt{N})$ vertices, the vertices of every such graph can be partitioned into three sets A, B, C such that no edges join a vertex in A with a vertex in B and the cost of A and B are each no more than two-thirds the cost of the graph.

To derive bounds to the planar circuit size of multilective circuits we extend the PST to a partition of the vertices of the planar graph into an arbitrary number of sets. This is done in two steps. A separator theorem is derived that applies under the simultaneous application of two cost functions. This is then applied to produce a P-partition separator theorem by successive refinement of partitions.

Lemma 1: Let $G = (V,E)$ be any N-vertex planar graph with vertex costs summing to no more than one. Then the vertices of G can be partitioned into three sets A_1, A_2, C such that no edge joins a vertex in A_1 with a vertex in A_2, $c(A_1), c(A_2) \leq (3/4)c(V)$, and the cardinalities of these sets satisfy $|A_1|, |A_2| \leq 5N/6$, and $|C| \leq K_2\sqrt{N}$ where $K_2 = 2\sqrt{2}(\sqrt{2/3}+1)$.

Theorem 1: Let $G = (V,E)$ be any N-vertex planar graph and $c: V \rightarrow \{0, 1\}$ a cost function on the vertices of G. Then the vertices of G can be partitioned into P disjoint sets $\{A_i \mid 1 \leq i \leq P\}$ each of cost

$$(1/4P)c(V) \leq c(A_i) \leq (4/P)c(V)$$

such for each A_i there is a separator set C_i of at most $K_3\sqrt{N}$ vertices and a set B_i of vertices, $B_i = V - A_i - C_i$, such no edge joins a vertex in A_i with a vertex in B_i. Here $K_3 = 2\sqrt{2}(\sqrt{2/3}+1)/(1-\sqrt{5/6})$.

3. BOUNDS ON MULTILECTIVE PLANAR COMPLEXITY

A framework given in [10] is extended to accommodate planar circuits that are multilective. Predicates and multiple output functions are treated.

Definition 2: A (one-output) function $f: X^n \rightarrow X$ is $w(u,v)$-*separated*, where $0 \leq u,v \leq 1$, if for any partition of its variables A into three sets U, V and Z such that $|U| \geq u|A|$, $|V| \geq v|A|$, there is some assignment $\mathbf{z}$ to the variables in Z such that for the subfunction in the remaining variables there exists at least $|X|^{w(u,v)}$ triples $\{(\mathbf{u}_i, \mathbf{v}_i, \mathbf{z})\}$ where $\mathbf{u}_i$ and $\mathbf{v}_i$ are over U and V, and such that $f(\mathbf{u}_i, \mathbf{v}_j, \mathbf{z}) = 1$ if $i = j$ and 0 otherwise. (We assume that input variables may be permuted to put them into this order.) A function $f: X^n \rightarrow X^m$ is also $w(u,v)$-*separated* if it contains a subfunction which is $w(u,v)$-separated.

The first result states that if a predicate is $w(u,v)$-separated then its multilective planar circuit size is quadratic in $w(u,v)$. It is established by demonstrating the existence of two subsets of the source nodes such that the number of associated variables in each that are not in the other is large. This is followed by the application of the extended Planar Separator Theorem.

Theorem 2: If $f: X^n \rightarrow X$ is $w(u,v)$-separated for $u = v = (1 - 16\mu/(P-1))/16\mu P$, then

$$C_P^{(\mu)}(f) \geq w(u,v)^2/4K_3^2$$

The next definition states a condition similar to those used by Thompson [12] and Brent-Kung [2] for multiple-output functions and one stated in [10] for semelective chip algorithms.

Definition 3: Let A and B be subsets of the sets of input and output variables, respectively, of a function $f: X^n \rightarrow X^m$. It is said to have a $w(u,v)$-*flow* if for all subsets $U \subset A$ and $V \subset B$ with $|U| \geq u|A|$, $|V| \geq v|B|$, $0 < u,v < 1$, there is some assignment to inputs not in U such that the resulting subfunction h of f from U to V has at least $|X|^{w(u,v)}$ points in the image of its domain. A function f also has a w(u,v)-*flow* if it contains a subfunction with this property.

The following theorem illustrates the importance of this definition. It also uses the extended Planar Separator Theorem.

Theorem 3: If $f: X^n \to X^m$ has a w(u,v)-flow for $u = (1-\mu/P)$ and $v = 1/4P$, then

$$C_P^{(\mu)}(f) \geq w(u,v)^2 / K_3^2$$

The application of these methods to predicates is illustrated by the following result.

Lemma 2: Let $\mathbf{x}_i$ denote a b-tuple of Boolean variables, and write $\mathbf{x}_i \simeq \mathbf{x}_j$ for $i \neq j$ if there is a cyclic shift of the first which is equal to the second. Let the Boolean predicate $f_C^{(p,b)}: \{0,1\}^n \to \{0,1\}$ on n = pb variables be defined as follows:

$$f_C^{(p,b)}(\mathbf{x}_1, \ldots, \mathbf{x}_p) = \begin{cases} 1 & \mathbf{x}_i \simeq \mathbf{x}_j, \text{ for some } i \neq j \\ 0 & \text{otherwise} \end{cases}$$

This predicate is w(u,u)-separated for

$$w(u,u) \geq (n/p)(u/(2-u))^2 - \log_2 2(n/p)$$

for $p = \lceil 2/u \rceil$ when $n \geq p\log_2(4p\log_2 4p)$.

Applying this lemma, Theorem 2 and the observation that any circuit that is multilective of order μn must have at least μn vertices, we have the following.

Theorem 4: The predicate $f_C^{(p,b)}: \{0,1\} \to \{0,1\}$ defined above has a multilective planar circuit size that is bounded below by

$$C_P^{(\mu)} = \Omega(max(n^2/\mu^{12}, \mu n)) = \Omega(n^{14/13})$$

when n is large.

This lower bound depends strongly on the multiplicative factor μ, in fact, much more strongly than the bounds given below for multiple output functions. This is the only known lower bound to the multilective planar complexity of a predicate.

Lower bounds are shown on the multilective planar complexity of a variety of multiple output functions. The following result demonstrates that (α,l,d)-independence, a generalization of a condition defined and used by Grigoryev [4] and others [11] to obtain lower bounds on space-time tradeoffs, can be used to derive lower bounds to planar circuit size. Functions for which such independence properties are known include the shifting function, binary integer multiplication, certain powers and reciprocal powers of binary integers, including the square root, binary integer division, transitive functions, and matrix multiplication.

Lemma 3: Let $f: X^n \to Y^m$ have sets *A* and *B* of input and output variables and let it contain a subfunction that is (α,l,d)-independent. Then it has a w(u,v)-flow given by

$$w(u,v) = \mathit{max}(\mathit{min}(vd,\ (l-(1-u)n)),\ 0)/\alpha.$$

The Discrete Fourier Transform (DFT) has been studied by Thompson [12] who has shown that $AT^2 \geq \Omega(n^2)$ for semelective algorithms. The following lemma states a bound on its w(u,v)-flow. It Grigoryev independence properties seem to be inadequate to use the above lemma.

Lemma 4: Let $f_{DFT}^{(n)}: X^n \rightarrow X^n$ denote the DFT over a finite ring X. Then it has a w(u,v)-flow which satisfies

$$w(u,v) = uvn$$

Lemma 3 is useful for bounding the w(u,v)-flow of many functions. However it is weak for matrix multiplication. Below we state a separate lower bound for this problem.

Lemma 5: Let the matrix product $C = D \times E$ of $s \times s$ matrices be defined over a ring with a set X of elements. Then it has a w(u,v)-flow for

$$w(u,v) = \mathit{max}(v-(1-u)^2,\ 0)s^2/2$$

for $0 < u,v < 1$.

Lemma 5 applies to matrix multiplication or functions which contain it as a subfunction. This includes the transitive closure of a 3n×3n matrix over a closed semiring and the inverse of matrix, as shown in [1]. However, a stronger result for matrix inversion can be obtained from the following identity in which it is shown that the inverse of a (2r)×(2r) matrix, two of whose submatrices P and Q are permutation matrices, computes the product of three matrices $P^{-1}BQ^{-1}$ which, as Vuillemin [14] has shown, is a transitive function of order r^2. Thus, matrix inversion has the independence properties of this transitive function.

$$\begin{bmatrix} P & B \\ 0 & Q \end{bmatrix}^{-1} = \begin{bmatrix} P^{-1} & -P^{-1}BQ^{-1} \\ 0 & Q^{-1} \end{bmatrix}$$

The bound on flows for matrix multiplication can be usefully applied to certain LUP decompositions. If a matrix A has such a decomposition with P the identity matrix, such as is true for A symmetric positive definite, and if the matrices L and U are of the form produced by the recurrences of Kung and Leiserson [7, p.281], then the matrix shown below has the indicated LU decomposition.

$$\begin{bmatrix} I & 0 & B \\ A & I & 0 \\ 0 & 0 & I \end{bmatrix} = \begin{bmatrix} I & 0 & 0 \\ A & I & 0 \\ 0 & 0 & I \end{bmatrix} \times \begin{bmatrix} I & 0 & B \\ 0 & I & -AB \\ 0 & 0 & I \end{bmatrix}$$

Consequently, this problem for 3n×3n matrices has a w(u,v)-flow which is at least that of the product of two n×n matrices. We summarize these results and those for functions whose Grigoryev independence properties are known.

Theorem 5: Let $f: X^n \rightarrow X^m$ be any of the following functions: the shifting function $f_s^{(n)}$, the binary multiplication function $f_m^{(n)}$, the DFT $f_{DFT}^{(n)}$, the binary functions $f_P^{(n,a)} = \left\lceil 2^{2n} x^a \right\rceil$ and $f_R^{(n,b)} = \left\lceil (2^n / y)^b \right\rceil$ where $a > 1$, and $b = q / 2^k > 0$, k and q integers independent of n, the (non-binary) transitive functions $h_G^{(n)}$ of order n, including s×s matrix inversion for $n = (s/2)^2$. Then,

$$C_p^{(\mu)}(f) = \Omega(max(n^2/\mu^2, \mu n)) = \Omega(n^{4/3})$$

Let $f_M: X^{2s^2} \rightarrow X^{s^2}$ denote s×s matrix multiplication over the ring X, let $f_{TC}: X^{s^2} \rightarrow X^{s^2}$ denote the transitive closure of an s×s matrix over a semiring X, and let $f_{LU}^{(s^2)}$ denote the Kung-Leiserson LU decomposition of s×s matrices for which one exists. Then for any of these functions

$$C_p^{(\mu)}(f) = \Omega(max(s^4/\mu^4, \mu n)) = \Omega(n^{6/5})$$

The lower bound on AT^2 that Kedem and Zorat [5] have derived for multilective chip algorithms for transitive functions can be translated into a lower bound of $\Omega(n^2/\mu^2)$ on multilective planar circuit size for this problem. The same is true for the bound on the function $x + y*z$ over a finite field that Yao [15] has derived.

4. MULTILECTIVE ALGORITHMS

The semelective FFT graph can be laid out on a grid in area $O(n^2 \log n)$ [13]. Using the construction of the type given by Goldschlager [3] this layout can be made planar with only a constant factor increase in area.

A multilective planar circuit for the FFT of size $O(n^{3/2})$ can be constructed as follows: Apply the decomposition of Savage and Swamy [9] and partition the FFT at the level which is at distance of (1/2)log n from the inputs. This decomposes the FFT into a set of $\sqrt{n}$ disjoint FFT's of $\sqrt{n}$ inputs above the cut, and a set of n trees below the cut, each reaching $\sqrt{n}$ inputs. Replicate the input trees for each such FFT, and replace the $O(p^2)$ crossings in each FFT graph on p inputs, $p = \sqrt{n}$, with a set of exclusive OR's, as suggested in [3]. The resulting circuit has area $O(n^{3/2})$.

Similarly, matrix multiplication can be realized by a planar circuit of size $O(n^{4/3})$ by computing each of the components of the result with separate circuits. Undoubtedly, more efficient planar algorithms result from more economical non-planar algorithms.

5. REFERENCES

1. A. V. Aho, J. E. Hopcroft, and J. D. Ullman, *The Design and Analysis of Computer Algorithms*, Addison Wesley (1974).
2. R. P. Brent and H. T. Kung, "The Area-Time Complexity of Binary Multiplication," *JACM* **28**(3) pp. 521-534 (July 1981).
3. L. M. Goldschlager, "The Monotone and Planar Circuit Value Problems are Log-Space Complete for P," *ACM Sigact News* **9**(2) pp. 25-29 (1977).
4. D. Yu. Grigoryev, "An Application of Separability and Independence Notions for Proving Lower Bounds of Circuit Complexity," *Notes of Scientific Seminars, Steklov Math. Inst.* **60** pp. 35-48 (1976).

5. S.M. Kedem and A. Zorat, "On Relations Between Input and Communication/ Computation in VLSI," *Procs. 22nd Ann. Symp. on Fnds. Comp. Sci*, pp. 37-41 (Oct. 28, 1981).
6. R. J. Lipton and R. E. Tarjan, "Applications of a Planar Separator Theorem," *SIAM J. Comput.* **9**(3) pp. 615-627 (Aug. 1980).
7. C. Mead and L. Conway, *Introduction to VLSI Systems*, Addison-Wesley (1980).
8. J. E. Savage, *The Complexity of Computing*, John Wiley and Sons (1976).
9. J. E. Savage and S. Swamy, "Space-Time Tradeoffs on the FFT Algorithm," *IEEE Trans. on Info. Th.* **IT-24** pp. 563-568 (Sept. 1978).
10. J. E. Savage, "Planar Circuit Complexity and the Performance of VLSI Algorithms," pp. 61-68 in *VLSI Systems and Computations*, ed. H. T. Kung, B. Sproull, G. Steele, Computer Science Press (1981).
11. J. E. Savage, "Space-Time Tradeoffs - A Survey," *Procs. Third Hungarian Conf. on Comp. Sci.*, pp. 93-104 (Jan. 26-28, 1981).
12. C. D. Thompson, "Area-Time Complexity for VLSI," *Procs. 11th ACM Ann. Symp. Th. Comp.*, pp. 81-88 (April 1979).
13. C. D. Thompson, "Fourier Transforms in VLSI," Report No. UCB/ERL M80/51, Electronics Research Lab, U. California at Berkeley (Oct. 15, 1980). Also appears in the Procs. 1980 IEEE Int'l Conf. on Circuits and Computers
14. J. Vuillemin, "A Combinatorial Limit to the Computing Power of VLSI Circuits," *Procs. 21st Ann. Symp. Foundations of Computer Science*, pp. 294-300 (Oct. 13-15, 1980).
15. A. Yao, "The Entropic Limitations on VLSI Computations," *Proc. 13th Ann. ACM Symp. on Theory of Computing*, pp. 308-311 (May 11-13, 1981).

HAMMING CODES, COMPUTER MEMORIES, AND THE BIRTHDAY SURPRISE

R. M. F. GOODMAN
University of Hull
Hull, England HU6 7RX

and

R. J. McELIECE
California Institute of Technology
Pasadena, California 91125

Abstract: We study the lifetimes of semiconductor RAM memories, with and without ECC. We find a close connection between the "coding gain" for such lifetimes, and the classical "birthday surprise" paradox of probability theory. One conclusion we reach is that if 1-ECC is used, error scrubbing is useless.

INTRODUCTION

All modern computer memories are built from VLSI RAM chips. Although individually these chips are highly reliable, they do fail occasionally, and when many chips are combined in a single large memory array, the overall system reliability may be unsatisfactory. For this reason, computer memories are often protected from certain combinations of chip failures by error-correction coding. The use of ECC often increases the system reliability dramatically, but even so, the system lifetime, as measured e.g. by the mean time between failures (MTBF), is finite. In this paper we shall develop mathematical techniques for estimating the MTBF for memory systems with and without ECC. For a given memory array and coding scheme, we will call the ratio of the coded MTBF to the uncoded MTBF the coding gain (CG). We shall see that in typical cases, CG's of the order of several thousand are not unusual. We shall also see that while the MTBF without ECC is dominated by failures of individual memory cells within a chip, with ECC present the MTBF is dominated by more complicated kinds of chip failure. One conclusion we draw from this is that "error-scrubbing," a technique by which "soft errors" can be purged from a memory system, is useless when ECC is present.

We imagine the memory is composed of M rows of n chips each. If a chip is an N x 1 RAM, then each row of chips equals N n-bit words of computer memory. We assume that each n-bit memory word is protected by an r-error correcting code (r-ECC), so that if r or fewer of these bits are in error, they will be corrected. (In current hardware, r=1 is the standard, but larger values of r are certainly feasible.) Each chip is subject to certain kinds of failures, e.g. single-cell, row, column, etc. If a given chip row has suffered a combination of failures which causes more than r bit errors in one of its component words, these errors cannot be corrected by the ECC, and a memory system failure will occur.

The authors would like to acknowledge financial support from the UK Science and Engineering Research Council, the Joint Services Electronics Program (contract N00014-79-C-0424), and the NATO Scientific Affairs Division (Belgium) (grant no. RG.170.81).

Supposing that the memory is in perfect condition at t=0, we denote by $N_i(t)$ the number of chip failures in the i-th row after t hours. Throughout this paper, we shall model $N_1, N_2, \ldots, N_m$ as independent, identically distributed Poisson processes. The i.i.d. assumption has been verified experimentally and we take it as an axiom. However, the Poisson assumption is a deliberate simplification of the situation. In the case of just one row, for example, a more accurate model (see [2] or [4]) is that individual chip failures are independent, where each chip has an identical a priori probability of $1-e^{-\lambda t}$ of having failed by time t. If there are n chips in the row, this means that the probability of exactly k chip failures after t hours is

$$\binom{n}{k}(1-e^{-\lambda t})^k \, e^{-\lambda t(n-k)} \quad , \tag{1}$$

whereas in our simplified Poisson model this is approximated by

$$e^{-\lambda nt} \, \frac{(\lambda nt)^k}{k!} \tag{2}$$

For the range of parameters relevant, we have concluded that (2) is a very accurate approximation to (1). For example, using (1) we calculate that the average waiting time until the k-th failure is

$$\frac{1}{n} + \frac{1}{n-1} + \ldots + \frac{1}{n-k+1} \quad ,$$

whereas using (2), this waiting time is simply k/n. In our applications, k will represent the number of chip failures in a row needed to produce a failure of the ECC. In such an application, k will be very much less than n, and the ratio of these two waiting times will be very close to 1.

Our calculations of memory system MTBF are based on the notion of the reliability function, which is the probability that a given device has not failed at time t. If R(t) denotes the reliability function, then the mean waiting time until failure is given by

$$\overline{T} = \int_0^\infty -R'(t).t\,dt$$

$$= \int_0^\infty R(t)dt.$$

If now R(t) denotes the reliability of a given row of chips, measured from the time (t=0) of the most recent failure, the reliablity of the M-row memory system will be $R(t)^M$, and we obtain the formula

$$\text{MTBF} = \int_0^\infty R(t)^M dt. \tag{3}$$

All of our results are based on Eq. (3). We now give a series of examples, arranged in order of increasing realism.

THE EXAMPLES

Example 1 (catastrophic chip failures): Here we assume only one kind of chip failure is possible, viz., complete failure of the entire chip. If an r-ECC is used, and if the intensity of the Poisson process in each row is denoted by λn, then

$$R(t) = e^{-\lambda nt} \sum_{k=0}^{r} \frac{(\lambda nt)^k}{k!} , \tag{4}$$

so that by (3),

$$\mathrm{MTBF} = \frac{1}{\lambda n} \int_0^\infty \left(e^{-x} \sum_{k=0}^{r} \frac{x^k}{k!} \right)^M dx . \tag{5}$$

Formula (5) represents the MTBF with ECC. Without ECC, there will be fewer chips per row, since no parity-check chips will be needed. If this number is denoted by k, then the MTBF without ECC is easily seen to be $(\lambda kM)^{-1}$, and so the coding gain is given by

$$\mathrm{C.G.} = \frac{k}{n} \cdot M \int_0^\infty \left(e^{-x} \sum_{k=0}^{r} \frac{x^k}{k!} \right)^M dx. \tag{6}$$

The factor k/n in (4) is the code rate. It represents the penalty, in system lifetime, caused by the necessary addition of $(n-k) \cdot M$ redundant chips to the system. The other factor, which we denote by B(m,r), turns out to have been studied under the name "birthday surprise number," in the literature of combinatorial probability theory. In classic statistics terminology, B(M,r) represents the average waiting time if a population containing M individuals is sampled with replacement until some individual has been selected r times. For example B(365,2) = 24.62. This means that on the average one need only stop about 25 people to discover two who have the same birthday.

In fact, we can prove that

$$B(M,r) = M \int_0^\infty \left(e^{-x} \sum_{k=0}^{r} \frac{x^k}{k!} \right)^M dx , \tag{7}$$

by considering computer memories! Simply think of each chip row as representing one day in the (M-day) "year", and a chip failure in row i as an individual whose birthday falls on "day i". Since each row is a Poisson process with indensity λn, the aggregate sequence of chip failures is a Poisson process with intensity λnM, and the times between successive chip failures are i.i.d. random variables $T_1, T_2, \ldots$, with common distribution function $1-e^{-\lambda nMt}$. The waiting time until some row has failed r times is thus

$$T = T_1 + T_2 + \ldots + T_N \quad , \tag{8}$$

where N is the number of chip failures (individuals to be interviewed) until some chip row has suffered r failures (some birthday has occured r times). By Walds' identity, [3, chapter 9]

$$\begin{aligned} \text{MTBF} &= E(T) \\ &= E(T_i) \cdot E(N) \\ &= \frac{1}{\lambda nm} \cdot E(N) \quad . \end{aligned} \tag{9}$$

But $E(N) = B(M,r)$, and the MTBF has already been computed in (5). Thus formula (7) does give the "birthday surprise" number, as advertised. (This formula was originally given in [5], where however the significance of the Poisson process to this problem was not evident).

For the rest of this paper, we shall specialize to r=1, the Hamming code. (We are not aware of any computer memory system that uses multiple error corrections.) If we expand the integral in (5) by the binomial theorem, and integrate term-by-term, we obtain the alternate formula

$$\text{C.G.} = R \cdot \left\{ 1 + 1 + \frac{M-1}{M} + \frac{(M-1)\,(M-2)}{M^2} + \ldots \right\} \tag{10}$$

this sum is easily evaluated for small values of M, and for large values of M the following asymptotic approximation should be used:

$$B(M,2) = \sqrt{\frac{\pi M}{2}} + \frac{2}{3} + \frac{1}{12}\sqrt{\frac{\pi}{2M}} - \frac{4}{135M} + \ldots \tag{11}$$

(Formula (11) can be derived by the general methods described by de Bruijn[1]. It is derived explicitly in Knuth[6], sec. 1.2.11.2.). The first two terms of (11) are accurate to within 1% already for M=8. What this shows is that the C.G. in example 1 is fairly modest unless there are very many rows of chips. (The C.G., using the median time between failures for example 1 was considered in [2]). We remark that formulas (10) and (11) can be used to obtain the numerical results of several of the examples obtained by elaborate numerical methods in ref. [4].

Example 2. Here we assume, more realistically, that individual memory cells within the chips fail independently. We find that the C.G. is again given by Eq. (4), except that now M represents the number of words in memory. For example if $M = 256K = 2^{18}$, by (5), the C.G. will be about $512\sqrt{\pi/2} = 642$, or slightly less, when the code rate k/n is accounted for.

Example 3. In this example we consider, besides individual memory cell failures, the effect of more complicated kinds of chip failure which actually do occur [4]. In a modern 256K RAM chip, for example, the 2^{18} individual memory cells are arranged in a 512x512 square array. It is possible for a chip component called a *column sense amplifier* to fail, in which case an entire column (512 memory locations) can fail simultaneously. Similarly if a *row driver* fails, an entire 512-bit row of memory locations will fail. And finally, though more rarely, a catastrophic failure within a given memory cell can cause both the *row and the column* containing that cell to fail simultaneously. In typical examples (see [4]) row and column failures account for perhaps only a few percent of the total number of chip failures. However, after some thought one can see that even with such a low percentage, the implication for memories protected by 1-ECC is that most memory system failures will be caused by a row failure in one chip and a column failure in another chip in the same row of chips. Thus a good approximation of the MTBF for memory systems protected by 1-ECC can be obtained by neglecting single-cell failures and considering only the row, column, and row/column failures.

We take as a model therefore an M-row computer memory in which the number of failures in each row is a Poisson process of intensity λn, and that the only kind of chip failures are "row", "column", and "row/column" failures. We denote these three events by A, B, and C, and their relative frequencies by a, b, and c. A system failure will occur when some row has suffered both a row error and a column error. Thus at time t, a given row will be intact *only* if it has suffered either: no failures at all; or only A failures; or only B failures; or exactly one C failure. It follows that the reliability function for one row of chips is (let $x = \lambda nt$)

$$\begin{aligned} R(t) &= e^{-x} && \text{(no failures)} \\ &+ e^{-x(b+c)}(1-e^{-ax}) && \text{(only A failures)} \\ &+ e^{-x(a+c)}(1-e^{-bx}) && \text{(only B failures} \\ &+ e^{-x(a+b)}(cx) && \text{(one C failure);} \\ &= e^{-x}(e^{ax}+e^{bx}-cx-1). \end{aligned}$$

Hence the MTBF is given by

$$\begin{aligned} \text{MTBF} &= \int R(t)^M dt \\ &= \frac{1}{n\lambda}\int_0^\infty \left\{ e^{-x}(e^{ax}+e^{bx}+(x-1) \right\}^M dx \quad , \end{aligned}$$

since the uncoded MTBF is $(\lambda km)^{-1}$, the coding gain in this case is

$$\text{C.G.} = R\cdot M \int_0^\infty \left\{ e^{-x}(e^{ax}+e^{bx}+(x-1) \right\}^M dx, \tag{12}$$

where R is, as above, the code rate. Calling the second factor in (12) $F(M;a,b,c)$, we have:

$$F(M;0,0,1) = B(M,2) \tag{13}$$

$$F(1:a,b,c) = \frac{1}{1-a} + \frac{1}{1-b} + c-1 \tag{14}$$

$$F(M;a,b,c) \sim \sqrt{\frac{\pi M}{2(1-a^2-b^2)}} + \frac{(a^3+b^3)-3(a^2+b^2)+2}{3(1-a^2-b^2)^2} \tag{15}$$

$$F(m;1/2,1/2,0) \sim \sqrt{\pi M} + 1 \quad . \tag{16}$$

Eq. (13) simply reflects the fact that if only failures of type C occur, any two failures in a chip row will cause a memory system failure - and this is the situation in Example 1.

This example can also be given a "birthday surprise" interpretation. If A denotes "boy" and B denotes "girl" (assume C=0), then $F(M;a,b,0)$ is the average number of children one must interview on a planet with any M-day year before finding a boy and girl with the same birthday. Assuming $a=b=1/2$, and comparing (11) and (16), we see that it takes about $\sqrt{2}$ times longer to get a "boy-girl" match than to get a unisex match. In particular with M=365, we see that on this planet one expects to interview about 35 candidates before finding the lucky couple. In our final example, we give a more practical application.

Example 4. To get a feeling for the results of example 3, consider the following numerical example. We imagine a memory consisting of M=4 rows of 64K RAM chips, each row containing k=32 data chips and 7 parity chips, with 1-ECC being implemented. This is a 1 megabyte memory array like that in the VAX 11/750, for example. Let us suppose that the chip failure rate is $\lambda = 10^{-5}$ per hour, with 99% of the failures being single-cell failures, and the other 1% being equally divided between row and column failures.

Without the ECC, the MTBF will be dominated by the single-cell failures. Assuming that the cell failures are independent, there are M = 4 x 64K rows and the failure rate per cell is $\lambda = 10^{-5}/64\text{K} = 1.5$ E-10. Hence

$$\text{MTBF (uncoded)} = \frac{1}{\lambda\cdot k\cdot M}$$

$$= 781 \text{ hours.}$$

$$\sim 1 \text{ month.} \qquad (17)$$

With 1-ECC, one is tempted to use formula (4) and conclude

$$\begin{aligned} \text{C.G.} &= \frac{32}{39} \cdot B(2^{18},2) \\ &= 0.82 \times 642 \\ &= 527, \text{ and so} \\ \text{MTBF (coded)} &= 527 \times 781 \\ &= 411 \text{ k hours} \\ &= \sim 48 \text{ years.} \end{aligned}$$

However, this is incorrect, since with the ECC, the MTBF is not dominated by single-cell errors, but rather by the combination of a row and a column failure within a single row of chips. Hence the relevant C.G. formula is (12), with $M=4$, $a = b = 1/2$, $c=0$, and $\lambda = 1\% \times 10^{-5} = 10^{-7}$. Since by direct calculation $F(4;1/2,1/2,0) = 4.66$, the C.G. is given by

$$\begin{aligned} \text{C.G.} &= \frac{32}{39} \times 4.66 \\ &= 3.82 \quad . \end{aligned}$$

However, this C.G. is with respect to the uncoded MTBF with $\lambda = 10^{-7}$, $M=4$, $k=32$, i.e., MTBF (uncoded) $= \frac{1}{\lambda k M} = 78$ kh. Thus we get

$$\begin{aligned} \text{MTBF (coded)} &= 3.82 \times 78 \text{ kh} \\ &= 298 \text{ kh} \\ &\sim 34 \text{ years .} \qquad (18) \end{aligned}$$

Hence, the actual C.G., obtained by comparing (17) and (18), is about 328. Although these numbers are mythical, they are typical, and they show e.g., if 1-ECC is used scrubbing is useless, since it is the unscrubbable row and column errors which then limit the system lifetime.

REFERENCES

1. de Bruijn, N. G., Asymptotic Methods in Analysis, Amsterdam: North-Holland, 1961.

2. Goodman, R. M. F., and R. J. McEliece, "Lifetime Analyses of Error-Control Coded Semiconductor RAM Systems," Proc. IEE, vol. 129, part E, No. 3, May 1982, pp. 81-85.

3. Hoel, P. G., S. C. Port, and C. J. Stone, Introduction to Probability Theory, Boston: Houghton-Mifflin, 1971.
4. Intel Corp, "Memory System Reliability with ECC," in: Intel Memory Applications Handbook, 1980.
5. Klamkin, M. S., and D. J. Newman, "Extensions of the Birthday Surprise," J. Combinatorial Theory 3 (1967), pp. 279-282.
6. Knuth, D. E., The Art of Computer Programming, vol. 1: Fundamental Algorithms, Reading, Mass.: Addison Wesley, 1973.

BOUNDS TO THE RATE-DISTORTION FUNCTION FOR REGENERATIVE COMPOSITE SOURCES[1]

MICHAEL J. CARTER and DAVID L. NEUHOFF
Computer, Information, and Control Engineering Program
University of Michigan
Ann Arbor, MI 48109

SUMMARY

Composite source models for speech are thought to offer improvement over autoregressive models for prediction of source code performance on actual speech data. A bound to the ultimate performance achievable by any code when applied to a composite source is the rate-distortion function R(D) for that source. However, the exact computation of R(D) for an arbitrary composite source remains an open problem.

In this work, we describe a method for obtaining upper and lower bounds to R(D) for *regenerative* composite sources. A regenerative composite source is characterized by a stationary switch process $\{S_j\}_{j=1}^{\infty}$ with alphabet $\{1, \ldots, N\}$ and a family of stationary processes $\{Z_j^{(i)}\}_{j=1}^{\infty}$, $i = 1,2, \ldots, N$, called *modes*. If the switch is in state i from time k to $k+m$, then the composite source output $X_k, \ldots, X_{k+m}$ has the same joint distribution as $Z_k^{(i)}, \ldots, Z_{k+m}^{(i)}$. Furthermore, $X_k, \ldots, X_{k+m}$ are conditionally independent of $(\cdots, X_{j-1}, X_j)$ and $(\cdots, S_{j-1}, S_j)$, given that $S_j \neq i$ for some $j<k$.

Our main result is that for regenerative composite sources,

$$f(D) \leq R(D) \leq f(D) + H(S) \tag{1}$$

---

[1] This work supported by the National Science Foundation under Grant ECS79-21075

where

$$f(D) = \lim_{k \to \infty} \inf_{\substack{\{d_{ij}\} \\ \sum_{i=1}^{N} \sum_{j=1}^{k} \alpha_{ij} d_{ij} \le D}} \sum_{i=1}^{N} \sum_{j=1}^{k} \alpha_{ij} \, R_{Z(i)}(j, d_{ij})$$

$$\alpha_{ij} = j \cdot Pr(S_1 \ne i, \, S_2 = i, \, \ldots, S_{j+1} = i, \, S_{j+2} \ne i)$$

and H(S) is the entropy rate of the switch process, $R_{Z(i)}(j, \cdot)$ is the j^{th} order rate-distortion function of the i^{th} mode process. We also show that the upper bound in (1) is valid for Fontana's *completely decomposable* composite sources [1]. A second result is that for regenerative composite sources,

$$R(D) \ge \inf_{\substack{\{d_i\} \\ \sum_{i=1}^{N} \beta_i d_i \le D}} \sum_{i=1}^{N} \beta_i R_{Z(i)}(d_i) \tag{2}$$

where $R_{Z(i)}(\cdot)$ is the RDF of the the i^{th} mode process, and $\beta_i = Pr(S_1 = i)$. The upper bound converges to the lower bound in the limit of ever "slower" switch processes, thus establishing (for regenerative composite sources) a result sought by Fontana. Bounds have also been obtained by Garde [2] in the case of a symmetric Markov switch process. Our method can accommodate arbitrary stationary switch processes (unlike Garde) and the bounds are valid for any particular switch process, rather than in the limit of slow switching as treated by Fontana.

---

[1] R.J. Fontana, "Limit Theorems for Slowly Varying Composite Sources," IEEE Trans. Inform. Theory, Vol. IT-26, pp. 702-709, Nov. 1980.

[2] S.A. Garde, "Communication of Composite Sources," Ph.D. dissertation, University of California, Berkeley, Berkeley, CA 1980.

A DECODING SCHEME FOR THE TERNARY GOLAY CODE

VERA PLESS
University of Illinois at Chicago, Chicago, Illinois

ABSTRACT

As the ternary (12,6) Golay code C has minimum weight 6, it can correct all errors of weights 1 or 2. We propose a decoding scheme for C which decodes all single and double errors after computing and analyzing at most 6 syndromes. These syndromes are with respect to two canonical bases of C. Any error of weight 3 or more is detected and our scheme locates a weight 3 coset leader in the coset where the error is. This would probably correct about a quarter of the weight 3 errors. If one merely wishes to detect these errors, the decoding is even simpler.

1. FACTS ABOUT C.

Our terminology is standard and can be found in [2] or [5]. There are several equivalent ways of describing this code. We suppose the coordinate indices are labeled $\infty',0',1',2',3',4',\infty,0,1,2,3,4$. We describe C by a generator matrix $H_1 = (I,A)$ where I is the 6×6 identity matrix and

$$A = \begin{pmatrix} 0&1&1&1&1&1\\ 1&0&1&2&2&1\\ 1&1&0&1&2&2\\ 1&2&1&0&1&2\\ 1&2&2&1&0&1\\ 1&1&2&2&1&0 \end{pmatrix}$$

Then [3] C also has the generator matrix $H_2 = (-A,I)$. It is clear from either generator matrix that C is self-dual so that both generator matrices are also parity check matrices.

It is well-known that the group of C is isomorphic to twice the Mathieu group M_{12} and has order 190,080. In [4], monomials are given for a subgroup of $2\cdot M_{12}$, $PGL_2(5)$, of order 120, which has the two sets of coordinate indices $\{\infty'$ thru $4'\}$ and $\{\infty$ thru $4\}$ as orbits. A further transformation, interchanging H_1 and H_2, exchanges these sets. For completeness we repeat these monomials here. In addition, we give one additional monomial, the scrambler, which together with the other monomials listed generates $2\cdot M_{12}$.

We first give the three monomials a), b), c) generating $PGL_2(5)$. Each is a permutation of the coordinate indices with possibly negating some of these.

a) $(0,1,2,3,4)(0',1',2',3',4')$

b) $(\infty,0)(1,4)(\infty',0')(1',4')(2,-2)(3,-3)$

In this monomial we perform the coordinate permutation $(\infty,0)(1,4)(\infty',0')(1',4')$, then negate the coordinates $2,3,2',3'$.

c) $(1,3,4,2)(1',-3',4',-2')(\infty,-\infty)(0',-0')$

Here, after the coordinate permutations $(1,3,4,2)(1',3',4',2')$, the columns $\infty,0',1',2',3'$ and $4'$ are negated. The next monomial interchanges the two sets and then negates one set, clearly exchanging H_1 and H_2.

d) $(\infty,-\infty',-\infty,\infty')(0,-0',-0,0')(1,-1',-1,1')(2,-2',-2,2')(3,-3',-3,3')$
$(4,-4',-4,4')$

We call the monomial s the scrambler as it scrambles up the two coordinate sets which have been either preserved or interchanged by a) - d).

s) $(\infty',3,4,1,2)(0',4',2',3',1')$

When the scrambler is added to a) - d), $2\cdot M_{12}$ is generated.

As C has minimum weight 6, it can correct all errors of weights 1 and 2. Hence C has a coset of weight 0, cosets of weights 1 and 2 and $3^6-(1+2\cdot 12+\binom{12}{2}2^2) = 440$ cosets of weight 3. So some errors of weight 3 can be corrected. If we choose any vector of weight 3, then x is contained in exactly 3 weight 6 vectors in C. This fact follows from Table 9.3, pg. 134 in [5]. Hence any coset of weight 3 contains exactly 4 vectors of weight 3. This accounts for all the vectors of weight 3 as $2^3\binom{12}{3} = 8.220 = 4.440$.

2. IF ERRORS OCCUR.

If x is a received vector, then we compute two syndromes, one with respect to H_1 called $S_1(x)$ or S_1 and one with respect to H_2 called $S_2(x)$ or S_2. An error of weight one has occurred iff one of these syndromes has weight one. If, $S_1 = \begin{pmatrix}0\\0\\1\\0\\0\\0\end{pmatrix}$, then the error vector $e = 0^2 1 0^9$ and $y = x - e$ is in C. An error of weight 2, with both error positions in one of the two sets, has occurred iff one of S_1 or S_2 has weight 2. If, say $S_2 = \begin{pmatrix}1\\0\\0\\2\\0\\0\end{pmatrix}$, then $e = 0^6 1 0^2 2 0^2$ and $y = x - e$ is in C.

Suppose a weight 2 error has occurred with an error position in each of the two sets. Sample S_1's and S_2's are given in Table 1 for this situation. Note first that this can only occur if both syndromes have weight 6 or one has weight 4 and the other has weight 5. Further, not all occurrences of the above description correspond to a weight 2 error. If the error positions are symmetric, $wt(S_1) = wt(S_2) = 6$ and the syndromes must either agree in exactly 5 positions or disagree in exactly 5 positions. All possibilities are described in Table 1.

If the error positions are not symmetric, then the syndromes must have weights 4 and 5 and have zeros in one pair of corresponding positions. There are many possibilities in this situation and all are described in Table 1.

3. THE DECODING SCHEME.

The purpose of this decoding scheme is to efficiently correct all weight 1 and weight 2 errors. Whenever more errors are found and a weight 3 coset leader in the error coset is found, that is added to the received vector. This should correct approximately one quarter of the weight 3 errors and can be omitted if it is not worth the effort.

Let x be the received vector. Store H_1 and H_2 and 6 transformations to be described later.

1. Compute S_1 and S_2 and their weights.
2. If $wt(S_1) = 1$ or 2, $e = S_1^T 0^6$. If $wt(S_2) = 1$ or 2, $e = 0^6 S_2^T$. We decode to $y = x - e$. This corrects all weight 1 errors and those weight 2 errors with both ones on a side.
3. If $wt(S_1)$ or $wt(S_2)$ is 3, we have detected more than two errors and located a weight 3 coset leader. Decode to $y = x - e$ where e is either $S_1^T 0^6$ (if $wt(S_1) = 3$) or $0^6 S_2^T$ (if $wt(S_2) = 3$). If $wt(S_1) = wt(S_2) = 3$, either will do.
4. We consider two situations here corresponding to condidates for weight 2 errors with the errors in the different halves.
   - a) $wt(S_1) = wt(S_2) = 6$ and S_1 and S_2 agree in exactly 5 positions or disagree in exactly 5 positions.
   - b) $wt(S_1) = 4$ or 5 and $wt(S_2) = 5$ or 4 respectively and S_1 and S_2 have zeros in a common postion.

If a weight 2 error occurs with the errors in different halves, then S_1 and S_2 must satisfy a) or b). However if S_1 and S_2 satisfy a) and b) it is also possible that more than 2 errors have occurred. Suppose that we do have a weight 2 error. Then in situation a) the error positions are symmetric and all possibilities are indicated in Table 1. A table lookup for correction is possible here. In situation b) the errors are not in symmetric positions and there are many possibilities (see Table 1). For case a) the location of the errors is in the position of agreement or disagreement. For case b), if the zeros occur in the i^{th} and j^{th} portions of S_1 and S_2, then the errors are in either positions (i,j') or (j,i'). For this reason we list monomials a_i, $0 \leqq i \leqq 4$, and one other needed monomial, b, in the group of C. Note that for each power of s, the ∞' coordinate moves from the left side to the right side and one other coordinate moves from the right side to the left.

We relabel the powers of s to indicate the coordinate being moved. Thus s^3 is labeled a_4 and moves coordinate 4 from one side to the other.

a_0: $(\infty',4,0)(0',4',2')(1,3,2)$

a_1: $(\infty',4,2,3,1)(0',2',1',4',3')$

a_2: $(\infty',3,4,1,2)(0',4',2',3',1')$

a_3: $(\infty',2,1,4,3)(0',1',3',2',4')$

a_4: $(\infty',1,3,2,4)(0',3',4',1',2')$

b : $(\infty,-3',-4',-1',-2')(0,4,2,3,1)$

$a_i^5 = b^5 = 1$, $i \neq 0$, $a_0^3 = 1$. If x is a vector and a is a monomial, we let xa denote the image of x under a.

Suppose 2 errors have occurred and we are in situation a). To decode, we take the appropriate a_i so that we have two error positions on one side. We then compute S_1 and S_2 for this new vector easily locating the errors. For example, suppose $S_1(x) = \begin{pmatrix}2\\1\\2\\1\\1\\2\end{pmatrix}$ and $S_2(x) = \begin{pmatrix}1\\1\\1\\2\\2\\1\end{pmatrix}$.

We see that the position of disagreement is the $(0',0)$ position, so we compute xa_0 which moves 0 to the left side and leaves $0'$ there.

Now $S_1(xt) = \begin{pmatrix}2\\0\\0\\0\\0\\1\end{pmatrix}$ so $e = (20^4 10^6)a_0^{-1} = (20^4 10^6)a_0^2 = (010^5 20^4)$ and we decode x to $y = x - e$.

If 2 errors have occurred and we are in situation b), we have two possibilities for the error positions. In this case we first move one of these positions to the other side and if we obtain a weight 2 syndrome, we decode. If we get a weight 3 syndrome, the assumption of 2 errors was wrong and we also decode. If neither of these happens, we move the second position to the other side. This is the situation in which we compute the most syndromes, six of them. Suppose for example that

$$S_1(x) = \begin{pmatrix} 2 \\ 1 \\ 0 \\ 1 \\ 2 \\ 2 \end{pmatrix} \text{ and } S_2(x) = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 1 \end{pmatrix}.$$

Here the possibilities for two errors are the $(\infty', 1)$ positions and the $(1', \infty)$ positions. To test the first possibility we use any a_i except a_1 which moves both ∞' and 1.

Consider xa_2. $S_1(xa_2) = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 2 \\ 2 \\ 0 \end{pmatrix}$ so $e = (0^3 1^2 0^7)a_2^4 = (0^3 1^2 0^7)a_3 = (10^7 10^3)$

and we decode x to $y = x - e$. If we had to test the second possibility, we could use transformation b).

Note that if we suspect we have two errors in symmetric positions (i', i), $i \neq \infty$, then we can use transformation a_i to decode. If $i = \infty$, any a_j will do. If we suspect that we have two errors in non-symmetric positions (i', j) or (j', i) and $i, j \neq \infty$, transformation a_j tests (i', j) and transformation a_i tests (j', i). If one of these, say i, is ∞ $(j \neq \infty)$, then any a_k, $k \neq j$, tests (∞', j) and b tests (j', ∞) if $j \neq 2$ and if $j = 2$, b^2 works.

If at any point we compute a weight 3 syndrome, we know that more than two errors have occurred and we have located a coset leader which can be used to decode. If we have detected an error of weight 3 or more but not a coset leader of weight 3, the a_i and powers of b can be used to find one.

5. COMMENTS AND ACKNOWLEDGEMENTS.

There is an infinite family of ternary symmetry codes [4] each of which has two generator matrices analogous to H_1 and H_2. The first five codes in the family have minimum weights as high as possible and it would be nice to have an efficient decoding method for all of them. The first symmetry code is the subject of this paper. The second code is a (24,12,9) code which can correct four of fewer errors. If a syndrome computed with respect to either the H_1 or H_2 matrices has weight 1, 2, 3, or 4, we can correct these errors. If not, we know that errors are distributed on both sides or more than four errors have occurred. Unfortunately, the group of this code is probably no bigger than $2 \times PGL_2(11)$, which either preserves or interchanges the sides, so a scrambler monomial in this group mixing up the sides is not possible. Perhaps other approaches can be used here.

We thank Robert Calderbank for helpful discussions. He told us he and John Conway have computed decoding permutations for the ternary Golay code. We used the CAMAC computer system [1] at the Computer Center of the University of Illinois in Chicago for various computations in M_{12}.

This work was supported in part by the National Science Foundation under Grant MCS-8201311.

REFERENCES

1. T. Grace and D. Bergstrand, "CAMAC Users Manual", University of Illinois at Chicago, 1981.
2. J. F. MacWilliams and N. J. A. Sloane, "The Theory of Error-Correcting Codes", North Holland, Amsterdam, 1977.
3. V. Pless, "On the uniqueness of the Golay codes", J. Comb. Theory, 5 (1968), 215-228.
4. V. Pless, "Symmetry codes over GF(3) and new five-designs", J. Comb. Theory, 12 (1972), 119-142.
5. V. Pless, "An Introduction to the Theory of Error-Correcting Codes", Wiley Interscience series in Discrete Math., 1982.

TABLE 1

SYNDROMES OF WEIGHT 2 ERRORS

The left column is the S_1 syndrome, the right column is the S_2 syndrome. The positions in error are given on top. As $S_i(-e) = -S_i(e)$, the negative of an error pattern is not given. As $(S_1(i'+j), S_2(i'+j)) = (S_2(i-j'), -S_1(i-j'))$, the pair of syndromes for the error pattern $-j'+i$ can be found from the $i+j$ pair. If $i,j \neq \infty$ and $j \geq i$, then the pair of syndromes for the error pattern for $i'\pm j$ can be obtained from the pair for the pattern $(0'\pm(j-i))$ by cycling the last 5 positions $(j-i)$ places from the bottom to the top. For these reasons all syndromes of weight 2 errors can be derived from the following.

Symmetric weight 2 errors				Non-symmetric weight 2 errors					
(∞´,∞)		(0´,0)		(∞´,0)		(0´,1)		(0´,2)	
1	2	1	1	2	0	1	1	1	1
1	1	1	2	0	0	2	0	0	0
1	1	1	1	1	1	0	0	1	1
1	1	2	2	2	1	1	2	0	1
1	1	2	2	2	1	2	2	1	2
1	1	1	1	1	1	2	1	2	1

SUBSTITUTION AND TRANSPOSITION CIPHERS

JAMES G. DUNHAM
Department of Electrical Engineering
Washington University
St. Louis, Missouri 63130

ABSTRACT

Substitution and transposition ciphers are two of the oldest cryptographic systems. In a general substitution cipher, or a homophonic cipher, each message letter corresponds to several substitute cryptogram letters, one of which is chosen at random for each message letter occurance. In a transposition cipher, the message is divided into groups of a constant length and a fixed permutation is applied to each group of message letters. A general theory is developed for substitution and transposition cipher systems under a ciphertext-only attack from an information theoretic point of view. A representation theorem is established which shows that a homophonic cipher system is equivalent to a fixed invertible randomization transformation followed by a simple substitution cipher system. For both substitution and transposition ciphers lower bounds on the key equivocation and the unicity distance are established in terms of the a priori key uncertainty and the redundancy rate of an appropriately defined expanded message source. In a matched homophonic cipher system the number of substitutes is proportional to the frequency of each letter. For a matched homophonic cipher and a transposition cipher these lower bounds are identical, suggesting equivalent cryptanalytic difficulty.

AN OPTIMAL APPROACH FOR TIME-DOMAIN TREE CODING OF STATIONARY GAUSSIAN SOURCES

BARUCH MAZOR and WILLIAM A. PEARLMAN
Electrical, Computer, and Systems Engineering Department
Rensselaer Polytechnic Institute
Troy, NY 12181

ABSTRACT

For discrete time, stationary, Gaussian sources and the squared error distortion measure, we derive an asymptotically optimal tree coding scheme. The scheme follows that of Berger [1], who constructed a tree code and proved the corresponding coding theorem for Gaussian autoregressive sources at rates above a so-called critical rate (R_C). We prove that the same scheme, with a modified derivation for the "coloring" filter used in the construction of the code, could be optimally applied to a large class of sources with no restriction on the coding rate. This was also substantiated via simulation results which indicate a performance level very close to the distortion rate function.

I. INTRODUCTION

Tree codes for sources with memory have recently attracted considerable interest. Berger [1] was the first to propose a tree coding scheme for autoregressive Gaussian sources under the squared-error distortion measure, valid in a region of low distortion values (below a so-called critical distortion). The scheme is rather attractive since it operates in the time domain with a simple tree structure. To construct the code, we first generate a tree with i.i.d. N(0,1) random variables residing on each branch. Then we feed each codeword, given by a sequence of branches through the tree, to a properly designed digital filter, thereby obtaining a so-called "colored" tree. Finally the "colored" tree is searched to obtain an appropriate reproduction codeword. Jakatdar and Pearlman [2], using a transform domain approach, proved that for stationary Gaussian sources there exist optimal codes which are confined to a generalized tree structure. The source output in this scheme, is transformed by the Karhunen-Leove (KL) transform, whereupon a search ensues for a suitable transform domain codeword.

In our paper, we generalize Berger's time domain scheme to accomodate a large class of stationary Gaussian sources encoded with arbitrary rate. The proof makes use of the random coding argument, and hence the marginal distribution of the reproduction process which solves the rate-distortion function R(D) is an essential parameter. We start (Section II) with the solution for this essential distribution, and then use it to construct the code and derive the appropriate coding theorem (Section III). Simulation results and the conclusion of the paper are presented in Sections IV and V respectively.

II. REPRODUCTION PROCESS STATISTICS

Let $\{X_t\}$ be a discrete-time Gaussian stationary source with the covariance sequence $\{\phi_n\}$ and power spectral density $S_x(\omega) = \sum_{n=-\infty}^{\infty} \phi_n \exp\{-jn\omega\}$. Assume $S_x(\omega)$ is in Lip 1 ($|S_x(\omega_1) - S_x(\omega_2)| \leq K|\omega_1-\omega_2|$, for some finite K>0 and all ω_1,ω_2). This condition on $S_x(\omega)$ guarantees the absolute summability of the covariance sequence $\{\phi_n\}$ ([3], Theorem, 4.2.5), and the ergodicity of the source.

Let $\underline{X} = \{X_t\}^N$ be a sequence of N observations from the source with the symmetric Toeplitz covariance matrix $\Phi_x = E[\underline{X}\ \underline{X}^T]$. Φ_x defines a unitary transform Γ^T which generates the decorrelated source sequence $\underline{X}' = \{X'_k\}^N$ as follows:

$$\underline{X}' = \Gamma^T \underline{X} \tag{1}$$

and:

$$E[\underline{X}'\ \underline{X}'^T] = \Lambda_x = \Gamma^T \Phi_x \Gamma = (\lambda_k \delta_{\ell k}) \tag{2}$$

where $\delta_{\ell k}$ is the Kronecker delta, and $\{\lambda_k\}$ is the set of eigenvalues associated with Φ_x. Since the source is Gaussian, $\{X'_k\}^N$ is an independent Gaussian sequence with $X'_k \sim N(0,\lambda_k)$, (normally distributed, mean 0, variance λ_k).

Let $\underline{Y}' = \{Y'_k\}^N$ be the transformed reproduction sequence. From the solution to $R_N(D)$ ([1], 4.5), we can relate $\underline{X}'$ and $\underline{Y}'$ using the "backward channel" relation:

$$\underline{X}' = \underline{Y}' + \underline{U}' \tag{3}$$

where $\underline{U}' = \{U'_k\}^N$ is a "noise" sequence, $\underline{U}'$ and $\underline{Y}'$ are statistically independent, and $\{U'_k\}^N$ is a statistically independent Gaussian sequence with the following diagonal covariance matrix:

$$E[\underline{U}'\ \underline{U}'^T] = \Lambda_u = (\min\{\theta,\lambda_k\}\ \delta_{\ell k}) \tag{4}$$

where $\theta = -1/2s$, and s is the slope of $R_N(D)$ at the point where it is evaluated. If we reapply Γ in (3) we obtain a similar relation in the time domain:

$$\underline{X} = \underline{Y} + \underline{U}$$

$$\underline{X} = \Gamma\ \underline{X}',\ \underline{Y} = \Gamma\ \underline{Y}',\ \underline{U} = \Gamma\ \underline{U}' \tag{5}$$

and:

$$E[\underline{U}\ \underline{U}^T] = \Phi_u = \Gamma\ \Lambda_u\ \Gamma^T \tag{6}$$

$\underline{Y}$ and $\underline{U}$ are statistically independent Gaussian sequences.

Now we use the following theorem to define the asymptotic statistic of the "noise" process $\{U_t\}$ and hence solve (5) for $\{Y_t\}$.

Theorem 1 [4, Thm. 4.2.1]

Let Φ_x be the covariance matrix of N observations from a stationary time series $\{X_t\}$ with an absolutely summable covariance sequence $\{\phi_n\}$ and a discrete power spectrum $S_x(\omega) = \sum_{n=-\infty}^{\infty} \phi_n \exp\{-jn\omega\}$. Define the diagonal matrix Λ whose diagonal elements are N uniformly spaced samples of the power spectrum $S_x(\omega)$, i.e.,

$$\Lambda = (S_x(2\pi k/N)\ \delta_{\ell k})\ k,\ell=0,1,\ldots,N-1.$$

The discrete Fourier transform (DFT) matrix,

$$F = N^{-1/2}(\exp\{-j2\pi k\ell/N\}),\quad k,\ell=0,1,\ldots,N-1,$$

asymptotically diagonalizes Φ_x to the diagonal matrix Λ. That is, for $\varepsilon>0$, there exists a number N_ε such that for all $N>N_\varepsilon$ every element of the matrix $F^*\Phi_x F - \Lambda$ has magnitude less than ε. ■

As a result we can conclude that for large N, the unitary transform Γ will coincide with the DFT, F, so that the corresponding diagonal covariance matrices Λ_x and Λ_u converge to:

$$\Lambda_x \rightarrow (S_x(k\ 2\pi/N)\quad \delta_{\ell k})\quad \ell,k = 0,1,\ldots,N-1 \tag{8}$$

$$\Lambda_u \rightarrow (\min\{S_x(k\ 2\pi/N),\theta\}\ \delta_{\ell k})\quad \ell,k = 0,1,\ldots,N-1 \tag{9}$$

Therefore $\{U_t\}$ is a Gaussian process which is asymptotically decorrelated by F to satisfy the following:

$$F^*\ \Phi_u\ F \rightarrow (\min\{S_x(k\ 2\pi/N),\theta\}\quad \delta_{\ell k}) \tag{10}$$

Suppose the Gaussian process $\{U_t\}$ possesses the power spectral density $S_u(\omega) = \min\{S_x(\omega),\theta\}$. Since $S_x(\omega)$ is in Lip 1, $S_u(\omega)$ is also in Lip 1 and hence possesses an absolutely summable covariance sequence and satisfies the necessary condition stated in Theorem 1. Therefore the process $\{U_t\}$ with spectrum $S_u(\omega)$ is a solution to (10).

Since $\{Y_t\}$ and $\{U_t\}$ are statistically independent we can define the reproduction process by:

$$S_y(\omega) = S_x(\omega) - S_u(\omega) = \max\{S_x(\omega) - \theta,\ 0\} \tag{11}$$

To generate a realization of $\{Y_t\}$ we pass an i.i.d. N(0,1) process $\{V_t\}$ through a linear time invariant system $H(\omega)$ with the following magnitude frequency response:

$$|H(\omega)| = (S_y(\omega))^{1/2} \tag{12}$$

$|H(\omega)|$ is well defined, and can always be very closely approximated by a realizable digital filter, regardless of the value of θ. (See [8]).

III. TREE CODE CONSTRUCTION AND CODING THEOREM

Following Berger's steps, for a given rate R choose a number α branches per node and a number β letters per branch, s.t. $R=\beta^{-1}\log\alpha$. Each codeword will be selected independently of the others according to the following procedure. First, construct a "white tree" with each branch filled with i.i.d. N(0,1) random variables. Next, feed each word from the white tree to the filter $H(\omega)$ in (12) and populate a "colored tree" (with the same parameters α, β), with these "colored" code words (see Fig. 1). By this process we generate a tree code ensemble, where a typical word from a typical code is distributed according to the optimum normal distribution of spectral density $S_y(\omega)$ as prescribed by the solution to R(D).

Given a sequence $\underline{X}$ from one source, the encoder searches the branches of the colored tree for the best available codeword, according to a chosen distortion measure, and forwards its path map to the receiver (Fig. 2a). As the receiver stores a replica of the encoder's tree, the received path map, assuming no channel errors, determines uniquely the correct reproduction codeword on the tree (Fig. 2b).

The encoder's search for the best available codeword proceeds through a discard criterion on m-tuples defined by

$$S(\delta,X^m) = \{Y^m: \rho_m(X^m,Y^m) \leq D(X^m) + m\delta\}$$

with

$$\delta>0 \ , \ \rho_m(X^m, Y^m) = \sum_{k=1}^{m} (X_k - Y_k)^2 \ ,$$

and $D(X^m)$ the conditional expected distortion given the source m-tuple X^m under the optimal test channel distribution of the rate-distortion function R(D). Following Davis and Hellman [5] the survival of the tree encoding process ensures a code and sequence with distortion approaching $E[D(X^m)]=D$ and is associated with the survival of an associated branching process with random environments (BPWRE). Their proof for the survival of an i.i.d. process must be generalized to our case of a stationary Gaussian process. Most of the steps involve the substitution of an m-tuple rate-distortion function and optimal test channel distribution. There are two crucial additional steps. The first is to prove the convergence in probability of the per-letter distortion $\frac{1}{N}\rho_N(\underline{X}, \underline{Y})$ to its mean D. The second involves the non-independence of successive generations of codeword m-tuples in the tree branching process. The next generation, however, is conditionally independent given the source state at that time. An argument in Berger [1, Sec. 6.3.4] shows how the source state can be specified as accurately as desired once every m source letters with only a negligible increase in the rate of the code. We invoke the same argument and refer the interested reader.

We show here only the first crucial step proving the convergence of $N^{-1}\rho_N(\underline{X}, \underline{Y})$ to D in probability.

$$N^{-1}\rho_N(\underline{X},\underline{Y}) = N^{-1} \sum_{k=1}^{N} (X_k-Y_k)^2 = N^{-1} \sum_{k=1}^{N} (X'_k - Y'_k)^2 = N^{-1} \sum_{k=1}^{N} (U'_k)^2$$

$$D = N^{-1} \ \overline{\rho_N(\underline{X},\underline{Y})} = N^{-1} \sum_{k=1}^{N} E\{(U'_k)^2\} = N^{-1} \cdot \sum_{k=1}^{N} \min\{\lambda_k \ \theta\}$$

$$\mathrm{Var}\ [N^{-1} \rho_N(\underline{X},\underline{Y})] = E\ [N^{-2} \sum_{k,\ell=1}^{N}\sum (U'_k)^2 (U'_\ell)^2] - D^2$$

$$= 2\,N^{-2} \sum_{k=1}^{N} [\min\{\theta,\lambda_k\}]^2 < 2N^{-1}\theta^2$$

Therefore:

$$\Pr[|N^{-1}\rho_N(\underline{X},\underline{Y}) - D|>\varepsilon]<\mathrm{Var}[N^{-1}\rho_N(\underline{X},\underline{Y})]/\varepsilon^2 < \frac{2\,\theta^2}{N\varepsilon^2} \xrightarrow[N\to\infty]{} 0$$

We now state the tree coding theorem as our conclusion.

Theorem 2 (Tree Coding Theorem)

Let $\{X_t\}$ be a discrete-time, Gaussian, stationary source with $S_x(\omega)\varepsilon$ Lip 1. Let $R(D)$ be the rate distortion function of $\{X_t\}$ with squared-error per-letter distortion measure. Then, given any $\varepsilon > 0$, there exists a tree code (as defined above) of sufficiently large blocklength with rate $R = \beta^{-1}\log_2\alpha$ and squared-error per-letter distortion less than $D(R) + \varepsilon$. ∎

IV. SIMULATION RESULTS

We performed several simulations by computer to evaluate the performance of the proposed tree coding scheme. Many different autoregressive (A.R.) Gaussian source models were employed at the low rates R = 1.0, 0.5 and 0.25 bits/sample.

An ℓ-th order A.R. source is described by:

$$X_t = \sum_{i=1}^{\ell} a_i X_{t-i} + Z_t$$

where $\{Z_t\}$ is a sequence of independent Gaussian random variates with variances $\sigma_z^2 = 1.0$. We will refer to a particular source by specifying its autoregression coefficients $(a_1, a_2, \ldots, a_\ell)$.

The coloring filters were designed using Deczky's program with a minimum squared error criterion [6]. When the number of poles and zeroes are specified, the program synthesizes a digital filter with a prescribed magnitude frequency response. The tree was searched using the M algorithm [7], and the performance is expressed in terms of the computed average distortion $\bar{D}$ (for ten blocks of 250 samples each) relative to the respective minimal attainable distortion $D(R)$ in dB, that is, $10\log_{10}(\bar{D}/D(R))$.

As a first step we examined the performance with respect to the order and structure of the "coloring" filter. We encoded different source models at different rates and in each case we applied several "coloring" filters which differ in their numbers of poles and zeroes. The results suggest that an optimal filter should consist of poles as well as zeroes and possesses a magnitude response which closely approximates $S_y(\omega)^{1/2}$ as defined in (12). At the same time, it was found that filters of an order equal or close to the source autoregression order are preferable. Tables 1a and 1b demonstrate a typical behavior for the sources designated V2:(1.5, -0.79, 0.07) and V5: (0.39, -0.9, 0.04) at rate 1.0. Using the best filters, we encoded the source V2 at rates 1.0, 0.5 and 0.25 bits/sample with search intensities M=16 and M=32. These rates are all below the source's critical rate of R_C=1.748. The performance obtained is given in Table 2, where we can observe the distortion $\bar{D}$ approaches $D(R)$ as the rate goes down. This trend is in agreement with the fact that at the limit of R=0, $\bar{D} = D(R) = \sigma_x^2$ is easily obtained with the single all-zero reproduction sequence.

Filter* / M: Search Intensity	2CZ 2CP	2CZ 1CP 1RP	1CZ 1CP 1RZ 1RP	3CZ 1CP 1RP	Source Poles
16	2.39	1.47	1.70	1.95	1.60
32	1.00	0.80	0.90	1.07	1.18

Table 1a: Performance in dB (10 log $\bar{D}/D(R)$) for source V2, R=1.0.

Filter* / M: Search Intensity	2CZ 1CP 1RP	2CZ 2CP	2CZ Source Poles	1RZ Source 1CZ Poles	Source Poles
8	1.10	1.15	1.14	1.08	1.25
16	0.72	0.75	0.73	0.71	0.96

Table 1b: Performance in dB (10 log $\bar{D}/D(R)$) for source V5, R=1.0.

* CP, CZ stands for complex pole and complex zero, respectively.
* RP, RZ stands for real pole and real zero, respectively.

M \ R	1.0	0.5	0.25
32	0.80	0.63	0.30
16	1.47	1.03	0.58

Table 2: Performance in dB for source V2.

V. CONCLUSION

We have proved an extension to Berger's tree coding scheme which applies to a large class of stationary Gaussian sources. Although tree coding schemes and theorems have existed for memoryless sources and for sources with some simple types of memory, there has been no constructive proof for sources with arbitrary types of memory. The solution suggests the use of a digital "coloring" filter with square magnitude response which approximates the power spectrum of the reproduction process $S_y(\omega)$. Although, in certain cases, this solution requires a filter of high order, simulation experiments indicate that a moderate order suffices. Also, simulation results for autoregressive Gaussian sources indicate a performance level in the vicinity of the rate distortion function and hence closely approaching the optimal performance.

REFERENCES

1. T. Berger, *Rate Distortion Theory.* Prentice Hall, 1971.

2. P. Jakatdar and W. A. Pearlman, "Very Low Rate Tree Coding of Stationary Gaussian Sources", *Proceedings of the Sixteenth Annual Conference on Information Sciences and Systems*, Princeton University, Princeton, NJ, March 1982, pp. 559-563.

3. C. S. Rees, S. M. Shah, and C. V. Stanojevic, *Theory and Applications of Fourier Analysis*. Marcel Dekker, 1981.

4. W. A. Fuller, *Introduction to Statistical Time Series*. John Wiley, 1976.

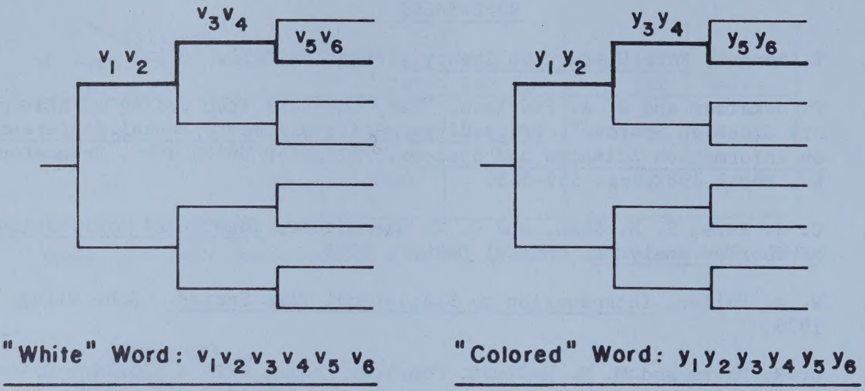
5. C. R. Davis and M. E. Hellman, "On Tree Coding with a Fidelity Criterion", IEEE Trans. Inform. Thy., IT-21, pp. 373-378, July 1975.

6. A. G. Deczky, "Synthesis of Recursive Digital Filters Using the Minimum P Error Criterion", IEEE Trans. on Audio Electroacoustics AU-20, Oct. 1972.

7. F. Jelinek, and J. B. Anderson, "Instrumentable Tree Encoding of Information Sources", IEEE Trans. Inform. Theory, IT-12, pp. 118-119, January 1971.

8. Berger restricts his development to the case of $\theta \leq \inf S_x(\omega)$, consequently limiting the average distortion to the range $D \leq D_C = \inf S_x(\omega)$ and the rate to $R \geq R_C = R(D_C)$. Here no such limitations in rate or distortion arise.

This research was sponsored by the Air Force Office of Scientific Research under Grant No. AFOSR-81-0188.

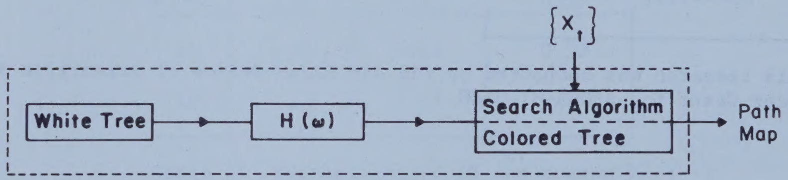

$v_1 v_2$
$v_3 v_4$
$v_5 v_6$
$y_1 y_2$
$y_3 y_4$
$y_5 y_6$
"White" Word: $v_1 v_2 v_3 v_4 v_5 v_6$
"Colored" Word: $y_1 y_2 y_3 y_4 y_5 y_6$

$$y_t = \sum_{\lambda=0}^{t} h_\lambda v_{t-\lambda}$$

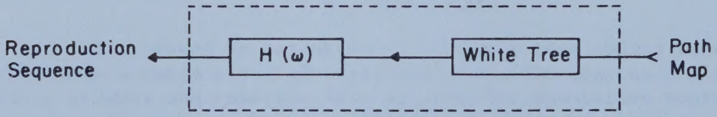
Path Map: 001

$\alpha = 2, \ \beta = 2, \ R = 0.5$

Figure 1. Correspondence between white and colored trees.


$\{X_t\}$
White Tree
$H(\omega)$
Search Algorithm
Colored Tree
Path Map

(a) Encoder


Reproduction Sequence
$H(\omega)$
White Tree
Path Map

(b) Decoder

Figure 2. Encoder and decoder for stationary normal time sequence $\{X_t\}$.

CAPACITY AND CUTOFF RATE OF FREQUENCY-HOPPED CHANNELS WITH SLOW RAYLEIGH FADING

WAYNE ERIC STARK
Department of Electrical and Computer Engineering
University of Michigan
Ann Arbor, Michigan 48109

ABSTRACT

This paper is concerned with the performance of codes on channels with slow Rayleigh fading. A slow frequency-hopped spread-spectrum system model with binary frequency-shift keying (FSK) modulation is assumed. The fading process is assumed to be constant over the duration of a hop and independent of the fading at all other hops. For this model we compute the capacity and cutoff rate when there are m bits transmitted per hop. The capacity and cutoff are computed when the receiver knows and does not know the fading level during each hop. We also compute the capacity and cutoff rate when diversity transmission is used. This is done for majority logic combining and square-law combining.

I INTRODUCTION

In this paper we will examine the asymptotic performance of codes as measured by the channel capacity and cutoff rate. The channel model we assume is that of a slow Rayleigh fading channel. The transmitter and receiver employ frequency-hopped spread-spectrum modulation in order to combat the fading. The fading process is assumed to be slow enough so that at any particular frequency the level of fade is a constant for the duration the frequency is used by the transmitter. Also the frequencies employed by the transmitter are such that the fading process is independent at different frequencies. This guarantees the channel is memoryless from hop to hop. Because of this we treat the frequency hopper and dehopper as part of the channel and consider the channel statistics over a single hop. The model we employ for the channel fits the general model of channels with memory in [2],[3].

Many communication systems subject to fading employ diversity transmission techniques to reduce the error rate. In addition some form of error-correcting coding may be used. In this paper we compute the capacity and cutoff rate for Rayleigh faded channels with diversity. We consider both square-law combining and majority logic combining. In section II we describe the channels models we will consider. In Section III the capacity and cutoff rate for slow Rayleigh fading channels with m bits per hop is computed. The capacity and cutoff rate with diversity is computed in section IV. Finally in section V we present some numerical results and draw some conclusions.

II CHANNEL MODELS

In this section the channel models for frequency-shift keying used in this paper are presented. We will only consider binary FSK so that the input alphabet is A={0,1}. Let X_j, $1 \le j \le m$, denote the input to the channel for the j-th bit of a hop, $X_j \in A$. Then $X_j = i$, $i \in A$, corresponds to transmitting a sinewave, s_i(t-jT), with power P for T seconds at frequency ω_i;

$$s_i(t) = (2P)^{1/2} \cos(\omega_i t + \vartheta_i) p_T(t), \quad 0 \le t \le T \tag{1}$$

where ϑ_i is the random phase of the i-th signal and is uniformly distributed between 0 and 2π. Also p_T(t)=1 for 0≤t≤T and is zero elsewhere. (We assume that ω_0, ω_1, and T are chosen so that s_0 and s_1 are orthogonal signals over the interval [0,T].)

For the Rayleigh fading channel, when the transmitted signal is $s_i(t-jT)$ the received signal is given by

$$r(t)=R(2P)^{1/2}\cos(\omega_i(t-jT)+\vartheta_i+\varphi)+n(t), \quad (j-1)T\leq t\leq jT \tag{2}$$

where R is a Rayleigh distributed random variable with mean square $2\sigma^2$, φ is uniformly distributed between 0 and 2π, independent of R, and n(t) is a white Gaussian noise process with two-sided spectral density $N_0/2$. The demodulator processes the received signal r(t) by computing (Y_{0j}, Y_{1j}) where

$$Y_{ij} = Y_{ijc}^2 + Y_{ijs}^2, \quad 0\leq i\leq 1 \tag{3}$$

$$Y_{ijc} = \int_{(j-1)T}^{jT} r(t)\cos\omega_i t\,dt \tag{4a}$$

and

$$Y_{ijs} = \int_{(j-1)T}^{jT} r(t)\sin\omega_i t\,dt \tag{4b}$$

A hard decision receiver the computes Y_j given by

$$Y_j = \begin{cases} 0 & \text{if } Y_{0j} > Y_{1j} \\ 1 & \text{if } Y_{1j} \geq Y_{0j} \end{cases} \tag{5}$$

An error occurs if Y_j is not equal to X_j The error probability $P_e(R)$ conditioned on the random variable R can be shown to be given by

$$P_e(R) = \frac{1}{2}\exp\left\{-R^2E/2N_0\right\} \tag{6}$$

When averaged over the random variable R the error probability P_e is given by

$$P_e = \frac{1}{2+\dfrac{\overline{E}}{N_0}} \tag{7}$$

where $\overline{E} = 2\sigma^2 E$ is the average received energy per transmission.

III CHANNELS WITH MEMORY

In this section we consider the case of a hard decision receiver so that the input and output alphabets are $A=\{0,1\}^m$. We compute the capacity and cutoff rate as a function of the memory length which is equal to the number of bits per hop. We also consider the case of the receiver knowing the value of the random variable R during the hop. This is called the channel with side information.

Since the channel is memoryless from hop to hop we can treat the channel with m bits per hop as a memoryless channel with input alphabet and output alphabet A^m. If the channel defined for each value of R is a symmetric channel [1] then the channel over the larger alphabet size is also symmetric. Since the channel is symmetric the capacity and cutoff rate are maximized by the uniform input distribution [1]. If $\underline{x} = (x_1,\ldots,x_m)$ is the input to the channel and has distribution $p(\underline{x})$ then the probability $p(\underline{y}\,|\,\underline{x})$ that $\underline{y} = (y_1,\ldots,y_m)$ is the output can be calculated when $\underline{x}$ and $\underline{y}$ differ in k places as

$$\alpha_{m,k} = p(\underline{y}\,|\,\underline{x}) = E\left\{P_e(R)^k(1-P_e(R))^{m-k}\right\}; \quad d(\underline{x},\underline{y})=k \tag{8}$$

where $d(\underline{x},\underline{y})$ is the Hamming distance between $\underline{x}$ and $\underline{y}$ and the expectation is with respect to the Rayleigh distributed random variable R. Note that $p(\underline{y}\,|\,\underline{x})$ depends only on the Hamming distance between $\underline{x}$ and $\underline{y}$. The expectation in (7) can be evaluated by expanding the factor $(1-P_e(R))^{m-k}$ as

$$\alpha_{m,k} = \sum_{i=0}^{m-k} \binom{m-k}{i} \frac{(-1)^{n-k-i})(½)^{n-i-1}}{2+\overline{E}/N_0(n-i)} \tag{9}$$

The capacity C(m) measured in bits per channel use is defined by

$$C(m) = \frac{1}{m} \max_{p(\underline{x})} \sum_{\underline{y}} \sum_{\underline{x}} p(\underline{x},\underline{y}) \log_2 \frac{p(\underline{y}\,|\,\underline{x})}{p(\underline{y})} \tag{10}$$

Since the uniform input distribution achieves capacity (10) becomes

$$C(m) = 1 + \frac{1}{m} \sum_{\overline{y}} p(\underline{y}\,|\,\underline{x}) \log_2 p(\underline{y}\,|\,\underline{x}) \tag{11}$$

When $d(\underline{x},\underline{y})$=k, $p(\underline{y}\,|\,\underline{x}) = \alpha_{m,k}$. There are $\binom{m}{k}$ such $\underline{y}$ for each $\underline{x}$. Thus

$$C(m) = 1 + \frac{1}{m} \sum_{k=0}^{m} \binom{m}{k} \alpha_{m,k} \log_2 \alpha_{m,k}. \tag{12}$$

The cutoff rate is defined as

$$R_0 = \frac{1}{m} \max_{p(\underline{x})} -log_2 \sum_{\underline{y}} \left[\sum_{\underline{x}} p(\underline{x}) \sqrt{p(\underline{y}\,|\,\underline{x})} \right]^2. \tag{13}$$

A similar derivation for the cutoff rate yields

$$R_0(m) = 1 - \frac{2}{m} \log_2 \sum_{k=0}^{m} \binom{m}{k} \sqrt{\alpha_{m,k}}. \tag{14}$$

These formulas can also be derived for M-ary channels (see [3]).

In some communication systems the receiver may know for each block of m bits what the value of the random variable R describing the fading level is. If coding is used the decoder can use this side information in the decoding process. We are interested in how much the capacity and cutoff rate change when side information is available. The capacity and cutoff rate can be calculated from results in [2]. When side information is available the capacity under certain circumstances is independent of the memory of the channel. For slow Rayleigh fading the capacity with side information is given by

$$\overline{C} = E\left\{ 1 + P_e(R) \log_2 P_e(R) + (1 - p_e(R)) \log_2 (1 - p_e(R)) \right\}. \tag{15}$$

The cutoff rate does depend on the memory length m and is given by .

$$\overline{R}_0(m) = 1 - \frac{1}{m} \log_2 E\left\{ \sum_{k=0}^{m} \binom{m}{k} [P_e(R)^k (1 - P_e(R))^{m-k}]^{½} \right\}^2. \tag{16}$$

The expectations in (15) and (16) must be evaluated numerically. It is a general result [2] that the capacity without side information increases to the capacity with side information as the memory length goes to infinity. Also the capacity without side information is less than the capacity with side information The cutoff rate without side information is less than the cutoff rate with side information. However, the cutoff rate with side information decreases as the memory length increases. See [2] for further discussion of these results.

IV CHANNELS WITH DIVERSITY

The performance of a memoryless (m=1) Rayleigh faded channel is often improved by the use of diversity or a repetition code. Diversity L just means that we transmit each symbol L times instead of 1 times. The L different code symbols of the code are sent at different frequencies so that each symbol experiences independent fading levels. Since the channel with m=1 is memoryless with 1 bit per hop we let Y_{ij} i=0,1 in (3) denote the receiver output for the j-th hop. Thus the channel output when diversity L is used is Y_{ij} i=0,1 j=1,2,...,L.

There are several ways to combine the Y_{ij} in (3) to form a channel output. With square-law combining the Y_{ij} j=1,2,...,L are added together for i=0 and 1. Let

$$Z_i = \sum_{j=1}^{L} Y_{ij} \tag{17}$$

A decision that 0 was transmitted is made if $Z_0 > Z_1$ and a decision that 1 was transmitted is made if $Z_1 > Z_0$. The channel output is just Z_i i=0,1. This channel is then a binary symmetric channel. The crossover probability $P_{e,L}$ is well known to be [4,eqn. 7.124b]

$$P_{e,L} = P_e^L \sum_{j=1}^{L} \binom{L+j-1}{j}(1-P_e)^j \tag{18}$$

where P_e is given in (7). The capacity and cutoff rate for this channel is found from

$$C(x) = 1 + x\log_2 x + (1-x)\log_2(1-x) \tag{19}$$

$$R_0(x) = 1 - log_2\left\{1 + 2\sqrt{x(1-x)}\right\} \tag{20}$$

where $x = P_{e,L}$

Another method of deciding what was transmitted is to make a decision on the basis of each received symbol Y_{ij} in (5) and then count the number of times a 0 was decided. Let W be the number of times $Y_j = 1$, i.e.

$$W = \sum_{j=1}^{L} Y_j \tag{21}$$

where Y_j is given in (5). The probability distribution of Y_j is given by

$$P\{Y_j = i \mid X = k\} = \begin{cases} P_e, & i \neq k \\ 1-P_e & i = k \end{cases} \tag{22}$$

so that conditioned on $X=k$, W is a binomial random variable with parameters P_e and L. If we let W be the channel output then it can be shown that the capacity and cutoff rate are

$$C(L) = 1 - \sum_{j=0}^{L} \binom{L}{j}(1-p)^j p^{(L-j)} \log_2\left(1 + \left(\frac{1-p}{p}\right)^{L-2j}\right) \tag{23}$$

$$R_0(L) = 1 - log_2(1 + [2\sqrt{p(1-p)}]^L) \tag{24}$$

where p=P_e. When W is the channel output we call this soft decisions. If on the basis of W we decide whether 0 or 1 was transmitted then the error probability $\overline{P}_{e,L}$ of this binary symmetric channel is for L odd just

$$\hat{P}_{e,L} = \sum_{j=\frac{(L+1)}{2}}^{L} \binom{L}{j} P_e^j (1-P_e)^{L-j} \tag{25}$$

The capacity and cutoff rate are then given by (19) and (20) respectively with $x = \hat{P}_{e,L}$.

V NUMERICAL RESULTS AND CONCLUSIONS

We have calculated the capacity and cutoff rate for the channels described in sections III and IV. What we are interested in knowing though is the required energy $\overline{E}_b$ per information bit to achieve some given performance level. When we are using codes of rate r, arbitrarily reliable communication is possible provide that

$$r < C(\bar{E}/N_0) \tag{26}$$

where $C(\bar{E}/N_0)$ is the channel capacity as a function of the signal-to-noise ratio. When using codes of rate r and diversity L the energy per bit is related to the energy per symbol by

$$\bar{E}_b = \bar{E}L/r \tag{27}$$

using (27) in (26) yields

$$\bar{E}_b/N_0 > LC^{-1}(r)/r \tag{28}$$

as the necessary condition for reliable communication. Similar calculations can be made for R_0 as well. In (26) to (28) r represents the rate of and outer code when used with the diversity. The overall code rate with diversity is the r/L. We will denote the minimum over r of the right hand side of (28) by min E_b/N_0 since this is the minimum bit signal-to-noise ratio needed to acieve capacity. The minimizing code rate will be denoted r_{opt}. The optimal overall code rate is then r_{opt}/L.

In Table 1 we list the values of min E_b/N_0 and the optimal code rate for channels with memory 1,5,8, and 15. Also listed is the same quantities when side information is avaiable. For Rayeigh fading side information can improve the performance by about 1.5 dB compared to Rayleigh faded channels without side information. In table 2 we list the values of min E_b/N_0 and r_{opt}/l for diversity with square-law combining. In Table 3 these values are listed for a 'soft decision' majority logic combiner while in Table 4 the corresponding values for hard decision majority logic combiner are given. We see that square-law combining with diversity 5 is 0.8 dB better than soft decision majority logic combining which is 1 dB better than hard decision majority logic combining.

REFERENCES

1. R.G Gallager, *Information Theory and Reliable Communication*. New York: Wiley, 1968.

2. R.J. McEliece and W.E. Stark, "Channels with Block Interference," submitted to *IEEE Trans. Information Theory*, Nov. 1981.

3. W.E. Stark, "Coding For Frequency-Hopped Spread-Spectrum Channels with Partial-Band Interference," Coordinated Science Laboratory Report R-945, (also Ph.D. Thesis) University of Illinois, July 1982.

4. J.M. Wozencraft and I.M. Jacobs, *Principles of Communication Engineering*. New York: Wiley, 1965.

	Capacity		Cutoff Rate	
Memory	min E_b / N_0 (dB)	r_{opt}	min E_b / N_0 (dB)	r_{opt}
m=1	10.2	0.21	12.9	0.13
m=5	9.9	0.22	13.5	0.11
m=8	9.7	0.23	13.9	0.08
m=15	9.5	0.24	14.6	0.05
m=1 (side information available)	8.5	0.24	10.8	0.21

Table 1. E_b / N_0 needed to acheive capcity and cutoff rate and the optimal code rate for channels with memory.

	Capacity		Cutoff Rate	
Diversity	min E_b / N_0 (dB)	r_{opt}	min E_b / N_0 (dB)	r_{opt}
L=1	10.2	0.21	12.9	0.13
L=3	10.5	0.13	12.8	0.11
L=5	10.9	0.09	13.1	0.08
L=7	11.3	0.07	13.4	0.06

Table 2. E_b / N_0 needed to acheive capcity and cutoff rate and the optimal code rate for channels with diversity L and square-law combining.

	Capacity		Cutoff Rate	
Diversity	min E_b / N_0 (dB)	r_{opt}	min E_b / N_0 (dB)	r_{opt}
L=1	10.2	0.21	12.9	0.13
L=3	11.1	0.11	13.3	0.10
L=5	11.7	0.08	13.7	0.08
L=7	12.1	0.06	13.9	0.06

Table 3. E_b / N_0 needed to acheive capcity and cutoff rate and the optimal code rate for channels with diversity L and soft decision majority-logic combining.

	Capacity		Cutoff Rate	
Diversity	min E_b / N_0 (dB)	r_{opt}	min E_b / N_0 (dB)	r_{opt}
L=1	10.2	0.21	12.9	0.13
L=3	12.0	0.11	14.5	0.08
L=5	12.7	0.08	15.0	0.06
L=7	13.1	0.06	15.4	0.05

Table 4. E_b / N_0 needed to acheive capcity and cutoff rate and the optimal code rate for channels with diversity L and hard decision majority-logic combining.

PARALLEL SPEECH ENCODING WITH MICROPROCESSORS

S. MOHAN
Department of Electrical and Computer Engineering
Wayne State University
Detroit, MI 48202

D.P. AGRAWAL
Department of Electrical Engineering
North Carolina State University
Raleigh, NC 27650

S.M. KOMANDUR
Department of Electrical and Computer Engineering
Wayne State University
Detroit, MI 48202

ABSTRACT

The stack algorithm, a sequential algorithm for source encoding, is currently being implemented using four microprocrssors connected in a shared system bus architecture. The Intel 8086 microprocessors are chosen for their fast cycle time, powerful instruction set, and suitability for multiprocessor applications. We have chosen off-the-shelf computers and components in order to reduce the time and cost involved in the development and to maintain system modularity. The parallel stack encoding algorithm suggested here is expected to reduce the computation time, and it can easily be modified to carry out sequential decoding. Compared with the stack algorithm performed on a single processor, the proposed parallel version will enhance the signal-to-noise ratio performance and will make real-time stack encoders a definite possibility.

1. INTRODUCTION

Digitally encoded speech transmission has attracted wide attention over the past decade. Sequential and convolutional schemes for encoding and decoding have been developed that utilize tree coding techniques [1,11, 12]. In a tree code, words may be mapped onto a perfectly regular tree structure with β symbols on each branch and b branches emanating out of each node. Each code word corresponds to a path through the tree, made up of these branches. Such schemes require effective tree codes and efficient searching algorithms [2,3]. Much effort has been devoted to obtaining algorithms for efficient and reliable transmission and for digital reproduction of speech [2,7,13,14]. We are particularly interested here in source encoders that convert analog/digital data into digital data in compressed form.

There are two factors in designing a source encoder; one is to find the best matched code and the other to develop an efficient search algorithm. The most commonly known search algorithms may be classified as metric-first (greedy), depth-first, and breadth-first schemes [8]. The metric-first schemes extend at any time the path with the best metric, metric being a measure of likelihood of successful search. The breadth-first search algorithms extend at once all branches that will ever be extended at a given level, whereas, the depth-first schemes extend a single path until the metric of the path being pursued falls below a discard criterion; they then backtrack to a node from which an unsearched branch exits and search this branch.

The multiple stack algorithm (MSA) devised by Chevillat and Costello [4] for sequential decoding has been implemented on the Zilog Z-80 microcomputer system [5]. Here we implement the stack algorithm using four Intel 8086, 16 bit microprocessors, interconnected using a shared bus system. It is expected that the computation time would be reduced, making it possible to implement real-time encoders.

2. REVIEW OF THE STACK ALGORITHM

A type of block code, which leads to instrumentable coding schemes, is the tree code. In a tree code the codewords cannot be chosen independently, but may be mapped onto a tree structure. A code tree as shown in Fig. 1, consists of a number of nodes arranged in levels. From each node at a given level b branches extend to the nodes at the next level. Associated with each branch of the tree are β branch letters chosen from the reproducing alphabet, which for our purposes is the binary alphabet. The sequence of branch letters encountered in following a path through the entire tree forms a code word. If the tree is L levels deep, the corresponding tree code has b^L code words, each of length $N = \beta L$. The corresponding rate, R, is given by $R = (\log_2 b)/\beta$ bits/source symbol. The aim of a source encoder search is to find a path with distortion as close as possible to Shannon's distortion rate function $\delta(R)$ [6]. This process is referred to as searching the code tree.

In the case of the stack algorithm, reference [7] considers the effect of the number of paths stored, S, the length of the paths, L, as well as a parameter D*, called the search bias on the performance of the algorithm and demonstrates that the stack algorithm achieves significant SNR gain over other methods.

Initialization

Initialization is done by clearing the memory table and creating an entry corresponding to the root node of the encoding tree.

Extension

Extension of a branch corresponds to calculation of the branch metric and storing the value in a list of size S.

The branch metric is given by $\mu(\hat{x}^{\ell\beta}) = \ell\beta D^* - d(x^{\ell\beta}, \hat{x}^{\ell\beta})$, where $x^{\ell\beta}$ represents $\ell\beta$ source symbols, $d(x^{\ell\beta}, \hat{x}^{\ell\beta})$ is the distortion incurred in representing the source symbols $x^{\ell\beta}$ by $\ell\beta$ reproducer symbols $\hat{x}^{\ell\beta}$. The code is defined as $\hat{x}(t) = \sum_i c_i q_{t-i}$, where q_i's are the quantizer output, and and c_i are the tap coefficients of the transversal filter [3]. The squared error distortion is $d(x,\hat{x}) = (x - \hat{x})^2$.

Deletion

Deletion is done by removing those paths from the list that have the worst metric in order to make room for new paths.

Symbol Release

When the length of the best path in the list equals the width of the list, the eldest M symbols of the best path are released to the user.

Ambiguity Check

The ambiguity check purges those paths in the list whose eldest symbols are not the same as the released symbols.

3. PARALLEL STACK ALGORITHM

All the selective search algorithms leave most paths unsearched. Only the Viterbi algorithm carries out exhaustive search and hence it is costly. The usual measure of efficiency for code searching algorithms has been the node computation, the number of branches visited during the progress of the scheme divided by the branches released as output, for a given level of source encoder fidelity or decoder probability of error. Several authors have found that using stack algorithm for source coding is exceptionally time consuming, even though it has the least node computation of any known method [8]. To choose the best algorithm for a given sitution, one must determine the combination of the parameters L, S and node computation that optimizes the cost function for the desired encoding distortion or error probability.

Unlike the single processor stack algorithm, the multiprocessor scheme proposed here will allow the user freedom to select the above parameters with less constraints. The SNR may be improved and the error probability reduced by a thorough search of the tree. But this would increase the computation time, and hence the cost. If we reduce the computation time by reducing the number of paths searched, the error rate will increase and will result in poor SNR. If parallelism is employed, we can simultaneously increase the number of paths searched (better SNR) and reduce the computation time (cost).

The basic steps that comprise the working of the parallel stack algorithm (PSA) are described in Figures 2 and 3. The PSA has two facets: one is the parallel searching by $N = 2^n$ processors, and the other the communication of information among processors leading to the selection of best S paths. Initially, the root node information is made available to all the processors. Processor 1 searches the sub-tree emanating from the first branch out of the root node, processor 2 the subtree from the second branch, and so on as shown in Fig. 2. Each processor performs the stack algorithm computation up to level L using the respective subtrees allotted to them, without releasing symbols and carrying out ambiguity checks.

Each of the N/2 processors, respectively, transfers their best S paths to their assigned N/2 neighbors so that concurrent transfer of data could be achieved without any conflict [9,10]. These processors merge the best S paths from their neighbors with their own and select best S paths. Out of these N/2, N/4 processors transfer their best S paths to the remaining N/4 processors in a predetermined order, for merging and selecting best S paths out of 2S in each one of them. The above steps are repeated until one processor has the best S paths out of NS paths.

The processor, which has the final best S paths, performs symbol release and ambiguity check. The best path information is then broadcast to the remaining N-1 processors. At this stage, each processor has a few best paths in their respective lists as shown in Fig. 4. At least one of these paths is L-M branches long. In the next cycle, each processor extends the best path as before until it finds a new best path of length L. Now merging, symbol-release, and ambiguity check are performed. This process is repeated until all the speech samples are encoded.

4. NETWORK ARCHITECTURE

The objective of our initial phase is to implement the parallel stack algorithm on a low cost architecture within a reasonalbe time period. Hence, we emphasized the use of off-the-shelf microcomputers and components which is expected to decrease the development cost and maintain the system modularity. A simple multiprocessor system with a shared system lens and a common memory is selected for the implementation of PSA.

A general block diagram of the network organization wth 4 computer module is shown in Fig. 5.

Procedure:

1. Start.
2. Initialize Root node.
3. Perform the stack algorithm in all the 4 processors without symbol release and ambiguity check on the sub-tree allotted to them.
4. Best S paths are transferred by P_2 and P_3 to the common memmory.
5. P_2 and P_3 interrupt P_1 and P_4 respectively.
6. P_1 and P_4 fetch S best paths of P_2 and P_3 from common memory respectively and each of them selects best S paths out of 2S paths.
7. Best S paths from P_4 are transferred to common memory and P_4 interrupts P_1.
8. P_1 fetches S paths of P_4 from common memory and selects S best paths out of 2S paths.
9. P_1 performs symbol release and ambiguity check.
10. P_1 transfers the best path information to common memory and interrupts P_2, P_5, P_4.
11. P_2, P_3 and P_4 fetch the best paths of P_1 from common memory.
12. Go to step 3 and repeat the cycle until all the speech samples are encoded.
13. END.

The versatile third generation INTEL 8086 based microcomputers are utilized for their fast instruction cycle time, instruction set efficiency, suitability for stack operation. The 8086 architecture also supports multiprocessor systems based on the concept of a shared system bus. All CPUs' in the system communicate with each other and share resources via the system bus. INTEL 8289 bus arbiter and 8288 bus controller are used to solve the priority and allocate the bus to the processor since the CPU request are asynchnonous. The code generator tables are stored in 4 EPROMS. The modularity of the system leaves room for further expansion.

5. CONCLUSION

In this paper we have reported on a software and hardware implementation of the stack algorithm using a multiprocessor network. Such a scheme is expected to reduce the computation time, and improve the SNR performance. Since no custom built equipment is used, the cost of hardware implementation is significantly low.

A similar network may be implemented at the receiving end for channel decoding. Other applications of this scheme are two dimensional tree coding of images, and contour extraction. Such an implementation makes real-time stack encoders a definite possibility.

REFERENCES

1. R.M. Fano, "A heuristic discussion of probabilistic decoding," IEEE Trans. Inform. Theory, Vol. IT-9, pp. 64-74, April 1963.
2. J.B. Anderson and J.B. Bodie, "Tree encoding of speech," IEEE Trans. Inform. Theory, Vol. IT-21, pp. 379-387, July 1975.
3. J.B. Anderson and C.W. Law, "Real-number convolutional codes for speech-like quasi-stationary sources," IEEE Trans. Inform. Theory, Vol. IT-23, pp. 778-782, Nov. 1977.
4. P.R. Chevillat and D.J. Costello, "A multiple stack algorithm for erasure free decoding of convolutional codes," IEEE Trans. Commun., Vol. COM-25, pp. 1460-1470, Dec. 1977.
5. H. Ma., "The multiple stack algorithm implemented on a Zilog Z-80 microcomputer," IEEE Trans. Commun., Vol. COM-28, pp. 1876-1882, Nov. 1980.
6. C.E. Shannon, "Coding theorems for a discrete source with a fidelity criterion, IRE Nat. Rec., March 1959.
7. S. Mohan and J.B. Anderson, "Speech encoding by a stack algorithm," IEEE Trans. Commun., Vol. COM-28, pp. 825-830, June 1980.
8. S. Mohan, "Analyses and cost evaluation of code tree search algorithms," Ph.D. Dissertation, McMaster University, Hamilton, Ontario, Canada, 1979.
9. M.A. Abidi and D.P. Agrawal, "On conflict-free permutations in multistage interconnection networks," Journal of Digital Systems, Vol. 4, No. 2, pp. 115-134, Summer 1980.
10. D.P. Agrawal and R. Jain, "A novel distributed processing scheme for computer analysis of motion," COMPCON Proc., pp. 614-621, Sept. 1980.
11. F. Jelinek, "A fast sequential decoding algorithm using a stack," IBM J. Res. Develop., Vol. 13, pp. 675-685, Nov. 1969.
12. J.B. Anderson, "A stack algorithm for source coding with a fidelity criterion," IEEE Trans. Inform. Theory, Vol. IT-20, pp. 211-226, March 1974.

13. N.S. Jayant and S.A. Christensen, "Tree encoding of speech using the (M,L)-algorithm and adaptive quantization," IEEE Trans. Commum., Vol. COM-26, pp. 1376-1379, Sept. 1978.

14. S.G. Wilson and S. Hussain, "Adaptive tree encoding of speech at 8000 bits/s with a frequency weighted fidelity criterion," IEEE Trans. Commun., Vol. COM-27, Jan. 1979.

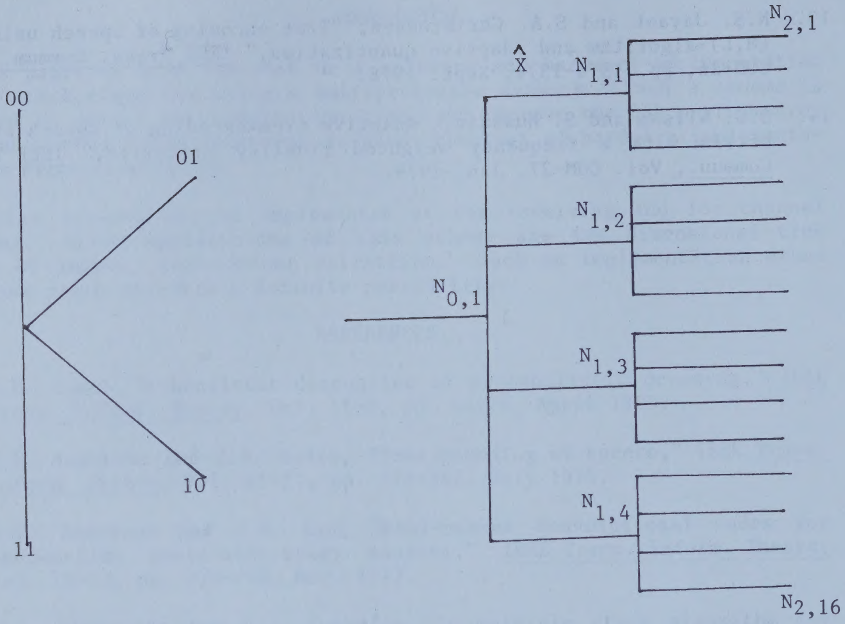

00
01
10
11
$N_{2,1}$
$\hat{X}$
$N_{1,1}$
$N_{1,2}$
$N_{0,1}$
$N_{1,3}$
$N_{1,4}$
$N_{2,16}$

Fig.1 CODE TREE. b = 4, B = 1

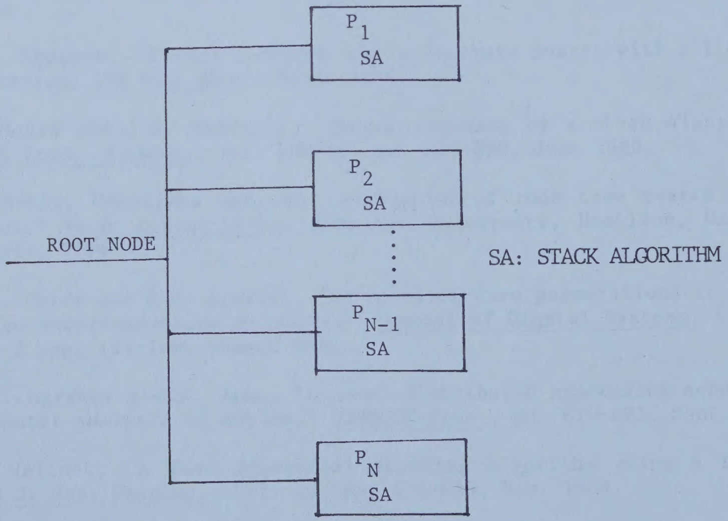

P_1
SA
P_2
SA
ROOT NODE
P_{N-1}
SA
P_N
SA
SA: STACK ALGORITHM

Fig.2 BLOCK DIAGRAM OF THE PARALLEL STACK ALGORITHM

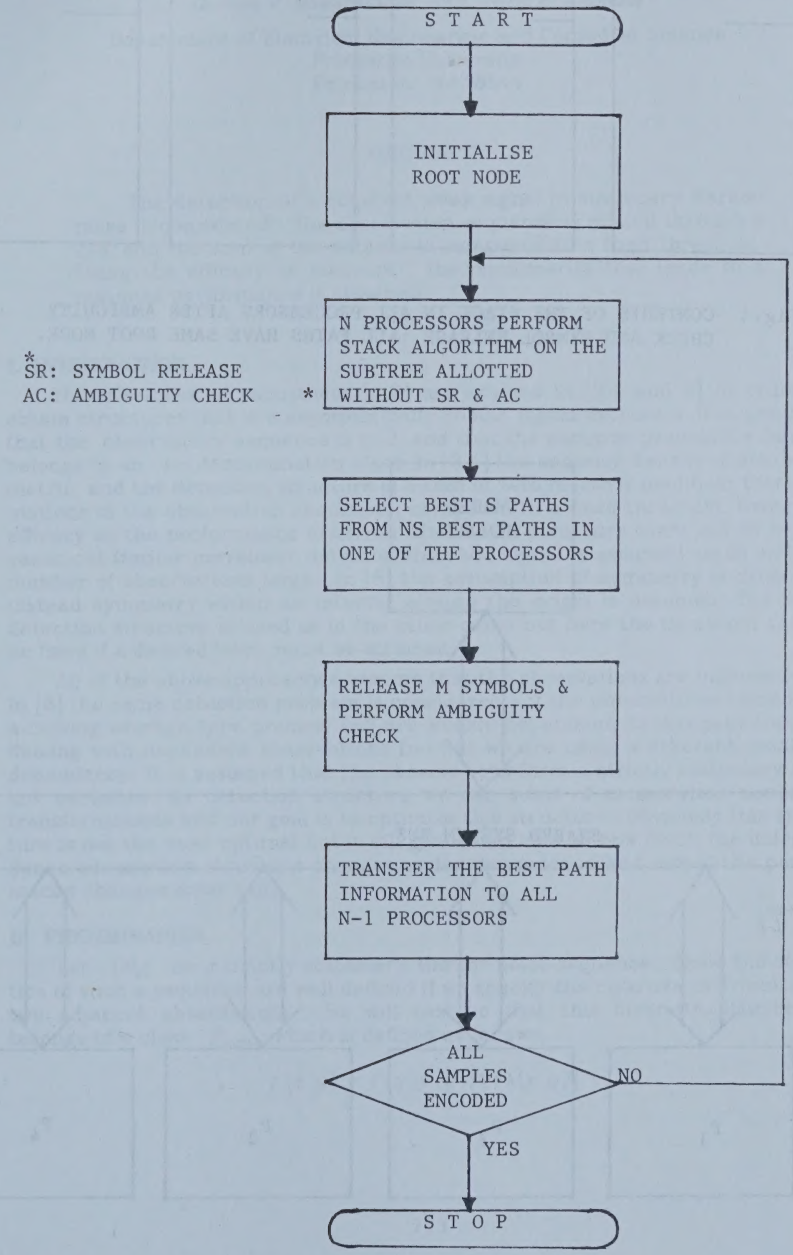

START
INITIALISE ROOT NODE
N PROCESSORS PERFORM STACK ALGORITHM ON THE SUBTREE ALLOTTED WITHOUT SR & AC *
* SR: SYMBOL RELEASE
AC: AMBIGUITY CHECK
SELECT BEST S PATHS FROM NS BEST PATHS IN ONE OF THE PROCESSORS
RELEASE M SYMBOLS & PERFORM AMBIGUITY CHECK
TRANSFER THE BEST PATH INFORMATION TO ALL N-1 PROCESSORS
ALL SAMPLES ENCODED
NO
YES
STOP

FIG.3 FLOW CHART OF THE PARALLEL STACK ALGOTITHM.

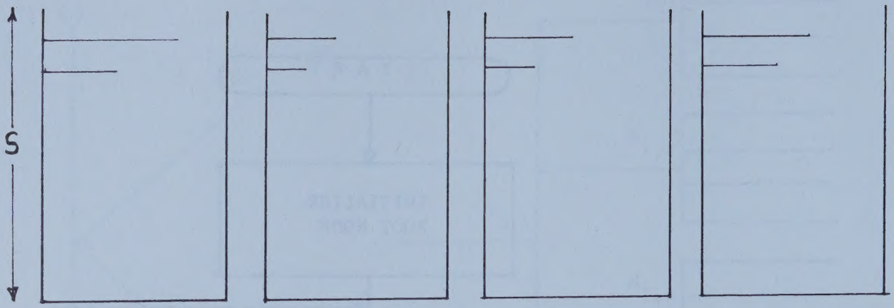


Fig.4 CONTENTS OF THE STACK IN ALL PROCESSORS AFTER AMBIGUITY CHECK AND SYMBOL RELEASE :ALL PATHS HAVE SAME ROOT NODE.

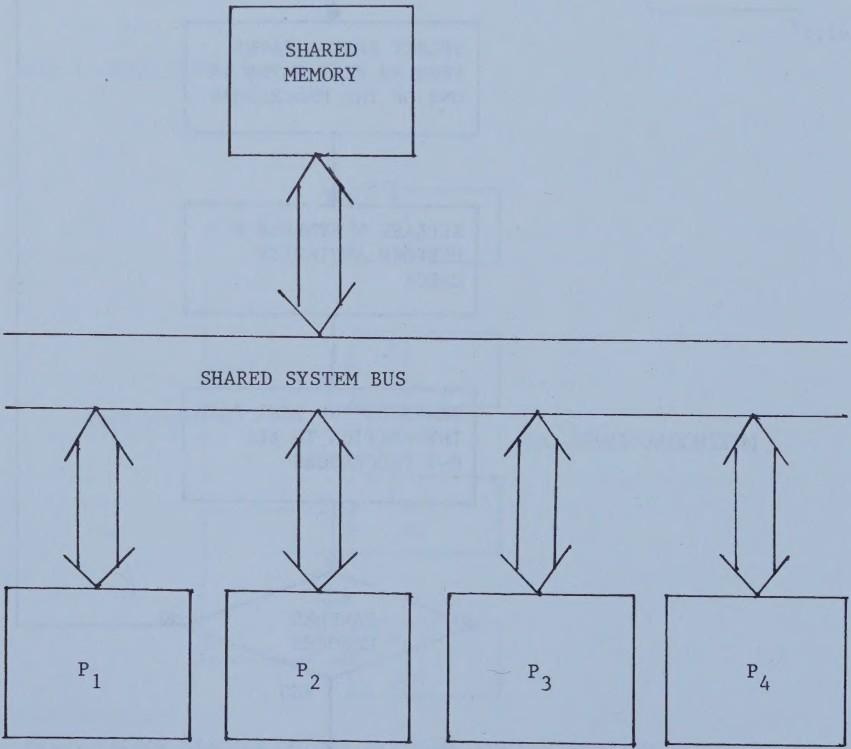

SHARED
MEMORY
SHARED SYSTEM BUS
P_1
P_2
P_3
P_4

Fig.5 ORGANISATION OF THE COMPUTER NETWORK.

MIN - MAX DETECTION OF A WEAK SIGNAL IN STATIONARY MARKOV NOISE

George V. Moustakides and John B. Thomas

Department of Electrical Engineering and Computer Science
Princeton University
Princeton , NJ 08544

ABSTRACT

The detection of a constant weak signal in stationary Markov noise is considered . The observation sequence is passed through a ZNL and the sum of the outputs is compared to a fixed threshold. Using the efficacy as measure , the nonlinearity that leads to a min-max performance is obtained.

I. INTRODUCTION

Huber's ideas of robustness [1,2] are applied in [3,4 and 5] in order to obtain structures that are asymptotically robust signal detectors. It is assumed that the observation sequence is i.i.d. and that the common probability density belongs to an $\varepsilon-$ contamination class. In [3,4] the common density is also symmetric and the detection structure is a sum of zero memory nonlinear transformations of the observation sequence , compared to a fixed threshold. Using the efficacy as the performance criterion ,the robust structure turns out to be the canonical *limiter correlator* detector,when the signal is assumed small and the number of observations large. In [5] the assumption of symmetry is dropped ; instead symmetry within an interval around the origin is assumed. The same detection structure is used as in the other cases but here the threshold cannot be fixed if a desired level must be attained.

All of the above approaches assume that the observations are independent . In [6] the same detection problem is considered but the observations come from a moving average type process and are weakly dependent. In this paper we are dealing with dependent observations too but we are using a different model of dependency. It is assumed that the observations form a strictly stationary Markov sequence. As detection structure we use sums of memoryless nonlinear transformations and our goal is to optimize this structure. Obviously this structure is not the most optimal but it will give us an idea on how much the independence assumption structure changes under dependency and also if the performance changes drastically.

II. PRELIMINARIES .

Let $\{N_i\}$ be a strictly stationary Markov noise sequence. Since the statistics of such a sequence are well defined if we specify the bivariate distribution of two adjacent observations , we will assume that this bivariate distribution belongs to a class $F_{\varepsilon,m}$ which is defined as follows

$$f(x,y) = f(x)f(y)\{1+\lambda(x,y)\} \tag{1}$$

$$\int_{-\infty}^{\infty}\lambda(x,y)f(x)dx = \int_{-\infty}^{\infty}\lambda(x,y)f(y)dy = 0 \tag{2}$$

$$f(x) = (1-\varepsilon)g(x)+\varepsilon h(x) \qquad 0 \le \varepsilon < 1 \tag{3}$$

$$|\lambda(x,y)| \le m < 1 \tag{4}$$

Condition (1) defines a common representation of a bivariate distribution. Condition (2) says that the function $f(x)$ is the marginal density. With Condition (3) we define an $\varepsilon-$ contamination model for the marginal. The density $g(x)$ is assumed to be known, symmetric, strongly unimodal and not equal to zero. The density $h(x)$ is unknown but symmetric and such that $f(x)$ has finite Fisher's information and the set $\{x : f'(x) = 0\}$ has $f-$ measure zero. Condition (4) limits our dependency model; notice that when $m = 0$ we are back to the i.i.d. case. The two constants ε and m are assumed to be known. As we will see shortly, Condition (4) is not only important because it makes $f(x,y)$ nonnegative but also because it makes the sequence $\{N_i\}$ a $\varphi-$ mixing sequence, a sufficient condition to guarantee asymptotic normality.

We now consider the detection of a constant signal; in particular, we would like to decide between the two hypotheses

$$\begin{aligned} H_0 &: \quad X_i = N_i \\ H_1 &: \quad X_i = N_i + s \qquad i = 1,2,\ldots \end{aligned}$$

where $\{X_i\}$ is the observation sequence and s a signal that tends to zero. Adopting the terminology from [5], as our detection structure we will use the *nonlinear-correlator* (NC) detector, which is of the form

$$T_n(x) = \sum_{i=1}^{n}\psi(x_i) \tag{5}$$

For the performance criterion we will use the efficacy. In order for the efficacy to exist and to be a valid criterion we have to impose restrictions on the nonlinearity $\psi(x)$, which will determine the allowable class Ψ of nonlinearities $\psi(x)$. We assume that $\psi(x)$ is an odd symmetric, zero mean, and second order function such that

$$\sigma_0^2(\psi) = E\{\psi(N_1)^2\} + 2\sum_{j=1}^{\infty}E\{\psi(N_1)\psi(N_{j+1})\} > 0 \tag{6}$$

We also assume that $\psi(x)$ satisfies conditions that are sufficient for the validity of the Pittman - Noether theorem. Such conditions are given in [7]. The assumption that $\psi(x)$ is odd symmetric is reasonable since the marginal densities are assumed even symmetric functions.

Proposition 1. Let $\{N_i\}$ be a strictly stationary Markov sequence with bivariate density $f(x,y)\in F_{\varepsilon,m}$; also let $\psi(x)\in\Psi$. Then $\{N_i\}$ is a $\varphi-$mixing sequence and $\sigma_0^2(\psi)$ defined in (6) is absolutely summable.

Proof. Because the sequence is stationary Markov, the bivariate density between N_1 and N_{j+1} is given recursively by

$$f_j(n_1,n_{j+1}) = \int_{-\infty}^{\infty} \frac{f_{j-1}(n_1,n_j)f(n_j,n_{j+1})}{f(n_j)}dn_j \qquad j=2,3,\ldots \tag{7}$$

where $f_1(x,y) = f(x,y)$. Now using induction and Equations (1,2,4,7) we can show that the density defined by (7) has the following form

$$f_j(x,y) = f(x)f(y)\{1+\lambda_j(x,y)\} \qquad j=2,3,\ldots \tag{8}$$

where $\lambda_j(x,y)$ satisfies equation (2) and is given by

$$\lambda_j(x,y) = \int_{-\infty}^{\infty}\lambda_{j-1}(x,z)\lambda(z,y)f(z)dz \qquad j=2,3,\ldots \tag{9}$$

and we define $\lambda_1(x,y) = \lambda(x,y)$. Also we have that

$$|\lambda_j(x,y)| \le m^j \tag{10}$$

Now let A_1 be an event from the σ-algebra generated by $\{N_1,N_2,\ldots,N_k\}$ and A_2 an event from the σ-algebra generated by $\{N_{k+i},N_{k+i+1},\ldots\}$. Since the set $\{N_{k+1},\ldots,N_{k+i-1}\}$ is not involved in the event $A_1 \cap A_2$ we have that

$$P(A_1\cap A_2) = \int_{A_1\cap A_2} \left[\frac{f(n_1,n_2)f(n_2,n_3)\cdots f(n_{k-1},n_k)}{f(n_2)f(n_3)\cdots f(n_{k-1})}\right]\times$$

$$\left[\frac{f_i(n_k,n_{k+i})}{f(n_k)f(n_{k+i})}\right]\times\left[\frac{f(n_{k+i},n_{k+i+1})\cdots}{f(n_{k+i+1})\cdots}\right] dn_1dn_2\cdots \tag{11}$$

Using (8,10) and noting that the first and the third term in (11) are the multivariate densities of $\{N_1,N_2,\ldots,N_k\}$ and $\{N_{k+i},\ldots\}$ respectively , we have

$$|P(A_1\cap A_2) - P(A_1)P(A_2)| \le m^i P(A_1)P(A_2) \tag{12}$$

So the sequence is symmetrically φ-mixing with $\varphi_i = m^i$. Clearly since $m < 1$ we have that $\varphi_i \to 0$ as $i \to \infty$ and also that $\sum_{i=1}^{\infty}\varphi_i^{1/2} < \infty$. And this takes care of the first part of Proposition 1.

To prove now that $\sigma_0^2(\psi)$ is absolutely summable , we use Equations (8,10) and we have

$$\begin{aligned}\sigma_0^2(\psi) &= \int_{-\infty}^{\infty}\psi^2(x)f(x)dx + 2\sum_{j=1}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\psi(x)\psi(y)f_j(x,y)dxdy \\ &\le \int_{-\infty}^{\infty}\psi^2(x)f(x)dx + 2\sum_{j=1}^{\infty}m^j \left[\int_{-\infty}^{\infty}|\psi(x)|f(x)dx\right]^2 \\ &= \int_{-\infty}^{\infty}\psi^2(x)f(x)dx + \frac{2m}{1-m}\left[\int_{-\infty}^{\infty}|\psi(x)|f(x)dx\right]^2 \end{aligned} \tag{13}$$

and , finally , using Schwartz inequality and that $\psi(x)$ is second order we have

$$\sigma_0^2(\psi) \leq \frac{1+m}{1-m} \int_{-\infty}^{\infty} \psi^2(x) f(x) dx \quad < \infty \tag{14}$$

and this proves the second part of Proposition 1.

Proposition 2 . Assume that $\{N_i\}$ is a noise sequence defined as above and that $E[\psi(N_1)] = 0$ and $E([\psi(N_1)]^2) < \infty$ where $\psi(x)$ is a measurable function. Then

$$\sum_{i=1}^{n} \psi(N_i)\, n^{-\frac{1}{2}} \xrightarrow{D} N\left(0, \sigma_0^2(\psi)\right) \tag{15}$$

Proof. It is a direct consequence of [8 , Theorem 21.1].

III. MIN - MAX DETECTION.

As our performance criterion we will be using the efficacy. Under the preceding assumptions it takes the following form

$$eff\,[\psi(x), f(x,y)] = \frac{[\int_{-\infty}^{\infty} \psi(x) f'(x) dx]^2}{\sigma_0^2(\psi)} \tag{16}$$

The problem we want to solve is the following. We would like to find a nonlinearity $\psi_r(x) \in \Psi$ and a density $f_r(x,y) \in F_{\varepsilon,m}$ such that

$$\sup_{\psi(x) \in \Psi} \; \inf_{f(x,y) \in F_{\varepsilon,m}} \; eff\,[\psi(x), f(x,y)] = eff\,[\psi_r(x), f_r(x,y)] \tag{17}$$

The first step is to try for a given $\psi(x)$ to minimize (16) over the density . Notice that the density depends on two functions, the marginal $f(x)$ and the function $\lambda(x,y)$. Since it makes no difference , we minimize over $\lambda(x,y)$ first. From (13) we have that

$$\sigma_0^2(\psi) \leq \int_{-\infty}^{\infty} \psi^2(x) f(x) dx + \frac{2m}{1-m} [\int_{-\infty}^{\infty} |\psi(x)| f(x) dx]^2 \tag{18}$$

Equality is achieved when $\lambda(x,y)$ is given by

$$\bar{\lambda}(x,y) = m \; sn_\psi(x)\, sn_\psi(y) \tag{19}$$

where $sn_\psi(x)$ is an odd symmetric function which for $x > 0$ is given by

$$sn_\psi(x) = \begin{cases} 1 & \text{if} \quad \psi(x) > 0 \\ -1 & \text{if} \quad \psi(x) < 0 \\ 1 \, or -1 & \text{if} \quad \psi(x) = 0 \end{cases} \tag{20}$$

Notice that the definition leads to a legitimate bivariate density since all conditions (1 to 4) are satisfied . Also applying (9) we have

$$\bar{\lambda}_j(x,y) = m^j \, sn_\psi(x) sn_\psi(y) \qquad j=1,2,\ldots \tag{21}$$

Thus if $\lambda(x,y)$ is given from (19) the efficacy becomes only a function of $\psi(x)$ and $f(x)$ and takes the form

$$eff^*[\psi(x),f(x)] = \frac{[\int_{-\infty}^{\infty}\psi(x)f'(x)dx]^2}{\int_{-\infty}^{\infty}\psi^2(x)f(x)dx + \frac{2m}{1-m}[\int_{-\infty}^{\infty}|\psi(x)|f(x)dx]^2} \tag{22}$$

In order to continue we have to minimize (22) over $f(x)$ and then maximize the result over $\psi(x)$. This is a min-max problem in itself with criterion function the eff^* given by (22). It turns out that this new problem has a saddle point ; in other words , there exists a pair $\psi_r(x)$ and $f_r(x)$ that satisfies the following saddle point relation

$$eff^*[\psi(x),f_r(x)] \le eff^*[\psi_r(x),f_r(x)] \le eff^*[\psi_r(x),f(x)] \tag{23}$$

for any allowable $\psi(x)$ and $f(x)$. As we know , any pair that satisfies (23) also satisfies a min-max relation like (17) , so the pair $\psi_r(x),f_r(x)$ is the one we are seeking. The left side inequality in (23) says that $\psi_r(x)$ is the optimum nonlinearity for $f_r(x)$, the one that will maximize the eff^* over the class Ψ The form of this optimum nonlinearity is given by the following theorem.

Theorem 1. Let $f(x)$ be a symmetric density function with finite Fisher's information and such that the set $\{ x : \psi_{lo}(x) = -\frac{f'(x)}{f(x)} = 0 \}$ has $f-$ measure zero . Then the nonlinearity $\psi_0(x)$ that maximizes (22) is given by

$$\psi_0(x) = \psi_{lo}(x) - \mu\varphi_0(x) \tag{24}$$

where $\varphi_0(x)$ is defined as

$$\varphi_0(x) = \begin{cases} \frac{1}{\mu}\psi_{lo}(x) & when \;\; -1 \le \frac{1}{\mu}\psi_{lo}(x) \le 1 \\ 1 & when \;\; 1 < \frac{1}{\mu}\psi_{lo}(x) \\ -1 & when \;\; -1 > \frac{1}{\mu}\psi_{lo}(x) \end{cases} \tag{25}$$

and μ is a constant that satisfies

$$S(\mu) = \mu + 2m \left[\frac{\int_{-\infty}^{\infty} f'(x)\varphi_0(x)dx}{1 - m + 2m\int_{-\infty}^{\infty}\varphi_0^2 f(x)dx} \right] = 0 \tag{26}$$

The proof of the theorem is in the Appendix . From Equations (24,25) we can see that the optimum nonlinearity $\psi_0(x)$ is closely related to the locally optimum $\psi_{lo}(x)$. Also the role of the function $\varphi_0(x)$ is to eliminate $\psi_{lo}(x)$ whenever it takes on values between $-\mu$ and μ . Now we are ready to define the pair that satisfies the saddle point. Since $\psi_r(x)$ is the optimum for $f_r(x)$ we need to define only $f_r(x)$

Theorem 2 . The density $f_r(x)$ that gives the solution to the saddle point problem is the following

$$f_r(x) = \begin{cases} (1-\varepsilon)g(x_1)e^{x_1(x+x_1)} & \text{for } x \leq -x_1 \\ (1-\varepsilon)g(x) & \text{for } |x| < x_1 \\ (1-\varepsilon)g(x_1)e^{-x_1(x-x_1)} & \text{for } x \geq x_1 \end{cases} \tag{27}$$

where $x_1 \geq 0$ and such that $f_r(x)$ has total mass equal to unity.

Proof . This density is nothing else than the one defined by Huber in [1,2]. It belongs to the $\varepsilon-$ contamination class with a legitimate density $h(x)$ (see [1]). To find $\psi_r(x)$ we apply Theorem 1 and if we denote by $\psi_{lo}^{r}(x)$ the locally optimum nonlinearity of $f_r(x)$, we get

$$\psi_r(x) = \begin{cases} 0 & \text{for } 0 \leq x \leq x_2 \\ \psi_{lo}^{r}(x) - \psi_{lo}^{r}(x_2) & \text{for } x_2 \leq x \leq x_1 \\ \psi_{lo}^{r}(x_1) - \psi_{lo}^{r}(x_2) & \text{for } x \geq x_1 \end{cases} \tag{28}$$

For $x \leq 0$ we recall that $\psi_r(x)$ is an odd function. The constant x_2 is defined as

$$\psi_{lo}^{r}(x_2) = \mu \tag{29}$$

and in order for (28) to be valid it has to satisfy $0 \leq x_2 \leq x_1$. In the Appendix we show that such an x_2 always exists.

Let us now for convenience define

$$M = \psi_{lo}^{r}(x_1) - \psi_{lo}^{r}(x_2) \tag{30}$$

Since $\psi_r(x)$ is nondecreasing we will have that $|\psi_r(x)| \leq M$. Up to now we have shown that $\psi_r(x)$ and $f_r(x)$ satisfy the left inequality in (23). To prove the right one is straightforward . If we define as $n(f)$ and $d(f)$ the numerator and the denominator of the eff^* then

$$n(f) = \left[\int_{-\infty}^{\infty} \psi'_r(x) f(x) dx\right]^2 = \left[(1-\varepsilon)\int_{-\infty}^{\infty} \psi'_r(x) g(x) dx + \varepsilon \int_{-\infty}^{\infty} \psi'_r(x) h(x) dx\right]^2$$

$$\geq \left[(1-\varepsilon)\int_{-\infty}^{\infty} \psi'_r(x) g(x) dx\right]^2 = n(f_r) \tag{31}$$

and

$$d(f) = \int_{-\infty}^{\infty} \psi_r^2(x) f(x) dx + \frac{2m}{1-m}\left[\int_{-\infty}^{\infty} |\psi_r(x)| f(x) dx\right]^2$$

$$\leq (1-\varepsilon)\int_{-\infty}^{\infty}\psi_r(x)^2 g(x)dx + \varepsilon M^2 + \frac{2m}{(1-m)}\left[(1-\varepsilon)\int_{-\infty}^{\infty}|\psi_r(x)|g(x)dx + \varepsilon M\right]^2$$

$$= d(f_r) \tag{32}$$

From (31) and (32) we see that $f_r(x)$ simultaneously minimizes the numerator and maximizes the denominator of eff^*, so that the pair $\psi_r(x)$ and $f_r(x)$ satisfies also the right inequality in (23). This completes the proof.

Summarizing the results , the solution to our min - max problem is the following : the density $f_r(x,y)$ is given by

$$f_r(x,y) = f_r(x)f_r(y)\{1 + m\ sn_{\psi_r}(x)\ sn_{\psi_r}(y)\} \tag{33}$$

where $f_r(x)$ is defined in (27) . The nonlinearity $\psi_r(x)$ is defined in (28) and these are all of the things we were looking for.

IV. NUMERICAL EXAMPLE .

Let the density $g(x)$ be the $N(0,1)$ normal. Then the $\psi_r(x)$ is

$$\psi_r(x) = \begin{cases} 0 & \text{for } |x| \leq x_2 \\ (|x|-x_2)sgn(x) & \text{for } x_2 \leq |x| \leq x_1 \\ (x_1-x_2)sgn(x) & \text{for } x_1 \leq |x| \end{cases} \tag{35}$$

For the density $f_r(x,y)$ we have the usual definition , only here the function $sn_{\psi_r}(x)$ can be equal to $sgn(x)$. In the following we give two tables, Table I contains the values of x_2 for different values of ε and m and Table II the ARE of $\psi_r(x)$ over $\psi_{lo}^r(x)$ when the underlying density is $f_r(x,y)$. Values for x_1 are not given . Since x_1 depends only on ε these values are the same with those given in [1] under the name k .

TABLE I

ε m	.1	.2	.3	.4	.5	.6	.7	.8	.9	1.0
.001	.147	.276	.400	.515	.635	.764	.912	1.096	1.373	2.630
.01	.145	.273	.393	.509	.626	.753	.895	1.069	1.320	1.945
.05	.139	.261	.374	.483	.591	.704	.826	.966	1.140	1.399
.10	.132	.247	.352	.452	.549	.648	.751	.861	.987	1.140
.15	.124	.232	.330	.422	.510	.597	.685	.776	.872	.980
.20	.117	.218	.309	.393	.472	.549	.625	.702	.780	.862
.30	.102	.190	.267	.337	.402	.463	.521	.577	.631	.685
.40	.087	.161	.226	.284	.337	.385	.430	.472	.512	.550
.50	.073	.134	.188	.235	.276	.314	.348	.380	.409	.436
.65	.051	.093	.130	.162	.190	.214	.236	.257	.274	.291
.80	.029	.054	.074	.091	.107	.121	.133	.143	.153	.162

TABLE II

ε m	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
0.001	1.01	1.03	1.07	1.12	1.19	1.29	1.42	1.64	2.04	4.48
0.01	1.01	1.03	1.06	1.11	1.18	1.26	1.38	1.55	1.85	2.68
0.05	1.01	1.03	1.05	1.09	1.14	1.21	1.29	1.39	1.54	1.77
0.1	1.01	1.02	1.05	1.08	1.12	1.17	1.22	1.29	1.38	1.49
0.15	1.00	1.02	1.04	1.07	1.10	1.13	1.18	1.23	1.28	1.35
0.2	1.00	1.01	1.03	1.06	1.08	1.11	1.14	1.18	1.22	1.27
0.3	1.00	1.01	1.03	1.04	1.06	1.08	1.10	1.12	1.14	1.16
0.4	1.00	1.01	1.02	1.03	1.04	1.05	1.06	1.08	1.09	1.1

APPENDIX.

Proof of Theorem 1. Notice that the value of the eff^* does not change if we multiply $\psi(x)$ by a constant. From one of the conditions for the validity of the Pittman - Noether theorem (Condition a) in [7]) we conclude that $\int_{-\infty}^{\infty} \psi(x) f'(x) dx < 0$. Thus maximizing Expression (22) is equivalent to maximizing

$$H(\psi) = -\int_{-\infty}^{\infty} \psi(x) f'(x) dx - \rho \left[\int_{-\infty}^{\infty} \psi^2(x) f(x) dx + \frac{2m}{1-m} \left[\int_{-\infty}^{\infty} |\psi(x)| f(x) dx \right]^2 \right] \quad (36)$$

where ρ is a Lagrange multiplier. We will show that (36) is maximized by

$$\psi_0(x) = \frac{1}{2\rho} \left[-\frac{f'(x)}{f(x)} - \mu \varphi_0(x) \right] \quad (37)$$

where μ and $\varphi_0(x)$ are defined in Equations (25,26). Let now $\psi_1(x)$ be any other nonlinearity from the class Ψ. Define the following variation

$$J(\gamma) = -\int_{-\infty}^{\infty} [(1-\gamma)\psi_0(x) + \gamma\psi_1(x)] f'(x) dx - \rho \left[\int_{-\infty}^{\infty} [(1-\gamma)\psi_0(x) + \gamma\psi_1(x)]^2 f(x) dx \right.$$

$$\left. + \frac{2m}{1-m} \left[\int_{-\infty}^{\infty} \{(1-\gamma)|\psi_0(x)| + \gamma|\psi_1(x)|\} f(x) dx \right]^2 \right] \quad (38)$$

where $\gamma \mathcal{E} [0,1]$. Notice that $J(0) = H(\psi_0(x))$ and $J(1) = H(\psi_1(x))$. By manipulating (38) we can rewrite it as

$$J(\gamma) - J(0) =$$

$$\gamma \int_{-\infty}^{\infty} \left[-f'(x) - 2\rho\psi_0(x) f(x) dx - \frac{4\rho m}{1-m} \left(\int_{-\infty}^{\infty} |\psi_0(x)| f(x) dx \right) \varphi_0(x) f(x) \right] [\psi_1(x) - \psi_0(x)] dx$$

$$-\gamma \frac{4\rho m}{1-m} \Big(\int_{-\infty}^{\infty} |\psi_0(x)| f(x) dx\Big) \bigg[\int_{-\infty}^{\infty} [\,|\psi_1(x)| - |\psi_0(x)|\,] f(x) dx$$

$$- \int_{-\infty}^{\infty} \varphi_0(x) [\psi_1(x) - \psi_0(x)] f(x) dx \bigg]$$

$$-\gamma^2 \rho \bigg[\int_{-\infty}^{\infty} [\psi_1(x) - \psi_0(x)]^2 f(x) + \frac{2m}{1-m} \{ \int_{-\infty}^{\infty} [\,|\psi_1(x)| - |\psi_0(x)|\,] f(x) dx \}^2 \bigg] \tag{39}$$

It is enough to show that $J(\gamma) - J(0) \leq 0$.

From the way that $\varphi_0(x)$ is defined in (25) we can see that

$$|\psi_0(x)| = \psi_0(x) \varphi_0(x) \tag{40}$$

$$|\varphi_0(x)| \leq 1 \tag{41}$$

By multiplying (37) by $\varphi_0(x) f(x)$ and integrating we can show , using (26) and (40) , that

$$\frac{2m}{1-m} [\int_{-\infty}^{\infty} |\psi_0(x)| f(x) dx] = \frac{\mu}{2\rho} \tag{42}$$

If we substitute (42) in the first term of the difference $J(\gamma) - J(0)$ and also use (37) , we get zero. The second term using (40) becomes

$$-\gamma \frac{4\rho m}{1-m} \Big(\int_{-\infty}^{\infty} |\psi_0(x)| f(x) dx\Big) \bigg[\int_{-\infty}^{\infty} \big[\, |\psi_1(x)| - \varphi_0(x) \psi_1(x) \big] f(x) dx \bigg] \tag{43}$$

and because of (41) the quantity in the brackets is nonnegative. Thus for $\rho > 0$, the above expression becomes nonpositive. The third term for $\rho > 0$ is clearly nonpositive too. And we have that the difference $J(\gamma) - J(0)$ is nonpositive and in particular $J(1) \leq J(0)$. If we also define $\rho = \frac{1}{2}$, Equation (37) becomes the same as (24).

For the existence of a μ that satisfies (26) notice the following : since by assumption $\psi_{lo}(x) \neq 0$ except on sets of $f -$ measure zero we have that as $\mu \to 0$ then $\frac{1}{\mu} \psi_{lo}(x) \to \pm\infty$ and thus $\varphi_0(x) \to sgn(\psi_{lo}(x)) = -sgn(f'(x))$. Thus from (26) we get

$$S(0) = -\frac{4m}{1+m} \int_{-\infty}^{\infty} |f'(x)| dx \leq 0 \tag{44}$$

Using Schwartz inequality it is easy to show that the integral in (44) is bounded by Fisher's information. As $\mu \to \infty$ the second term in $S(\mu)$ remains bounded so that $S(\mu) \to +\infty$. By continuity , there exist a μ that satisfies

$S(\mu) = 0$. And this completes the proof of Theorem 1.

Existence of x_2 . To prove that there exists an x_2 that satisfies $x_2 \le x_1$ for the density defined in (27) it is enough to show that $S(\psi_{lo}^T(x_1)) \ge 0$. Notice that when $\mu = \psi_{lo}^T(x_1)$ then

$$\varphi_0(x) = \frac{\psi_{lo}^T(x)}{\psi_{lo}^T(x_1)} \tag{45}$$

On applying this in the expression for $S(\mu)$ and manipulating we get

$$S(\psi_{lo}^T(x_1)) = \psi_{lo}^T(x_1) \left[1 - \frac{2mFI}{[\psi_{lo}^T(x_1)]^2 (1-m) + 2mFI} \right] \ge 0 \tag{46}$$

where by FI we denote Fisher's information. Thus the solution to $S(\mu) = 0$ must be less than $\psi_{lo}^T(x_1)$ and since $\psi_{lo}^T(x)$ is a nondecreasing function we get $x_2 \le x_1$. And this completes the proof.

ACKNOWLEDGEMENT .

This research was supported by the U S Army under Grant DAAG 29-82-K-0095 and by the National Science Foundation under Grant ECS-79-18915 .

REFERENCES .

[1] P.J. Huber , " Robust estimation of a location parameter " *Ann. Math. Stat.*, vol 35 , pp. 73-101 ,1964.

[2] P.J. Huber , " A robust version of the probability ratio test," *Ann. Math. Stat.*, vol 36 , pp. 1753-1758 , 1965 .

[3] R.D. Martin and S.C. Schwartz , " Robust detection of a known signal in nearly Gaussian noise," *IEEE Trans. Inf. Theory* , vol. IT-17 , pp. 50-56 , Jan. 1971 .

[4] S.A. Kassam and J.B. Thomas , " Asymptotically robust detection of a known signal in contaminated non-Gaussian noise," *IEEE Trans. Inf. Theory* , vol. IT-22 , pp. 22-26 , Jan. 1976.

[5] S.A. Kassam , G.V. Moustakides and J.G. Shin , " Robust detection of known signals in asymmetric noise," *IEEE Trans. Inf. Theory* , vol. IT-28 , pp. 84-91, Jan. 1982.

[6] H.V. Poor , " Signal detection in the presence of weakly dependent noise , Parts I and II," *IEEE Trans. Inf. Theory* , vol IT-28 , pp. 735-752 , Sept. 1982.

[7] D.R. Halverson and G.L. Wise , " Discrete time detection in $\varphi-$ mixing noise," *IEEE Trans. Inf. Theory* , vol. IT-26 , pp. 189-198 , March 1980.

[8] J. Billingsley , Convergence of probability measures. New York : Wiley , 1968, pp. 166-195.

M-STAGE NONPARAMETRIC RANK DETECTORS WHICH REDUCE THE AVERAGE SAMPLE SIZE AND RANKING TIME

SAWASD TANTARATANA

Department of Electrical Engineering
Auburn University
Auburn, Alabama 36849

Abstract

M-stage nonparametric detectors using rank test statistics are proposed. An M-stage Wilcoxon detector is studied in detail. In each (except the last) stage, the test statistic is compared to two thresholds. A signal is declared if the upper threshold is crossed or a signal-absent is decided if the lower threshold is crossed; otherwise, the test goes on to the next stage. Data ranking is performed among samples in each stage. With respect to a fixed-sample-size Wilcoxon detector of comparable power function, reduction in the average sample size is observed, in addition to reduced data ranking time.

I. Introduction

Let $x_1, x_2, \ldots,$ be the realization of the observed random variables $X_1, X_2, \ldots,$ we wish to test a hypothesis H_0 versus an alternative H_1 under the assumption that $X_1, X_2, \ldots$ are independent and identically distributed, where

$$\begin{aligned} H_0 &: X_i \sim f(x) \quad , \quad f(x) = f(-x) \\ H_1 &: X_i \sim f(x-\theta) \quad , \quad \theta > 0 \quad . \end{aligned} \tag{1}$$

Here, the actual value of θ and the functional form of $f(x)$ are not available, in which case a nonparametric detector is needed.

Before describing an M-stage rank detector, we include fixed-sample-size (FSS) linear and rank detectors here for convenience. A FSS linear detector compares the sum of the observations to a threshold, namely,

$$\sum_{j=1}^{n} x_j \begin{cases} > t \rightarrow \text{say } H_1 \\ = t \rightarrow \text{say } H_1 \text{ with prob. } \mu \\ \text{otherwise} \rightarrow \text{say } H_0 \end{cases} \tag{2}$$

where n is the sample size or the (fixed) number of observations, t is the threshold and μ is the randomization probability. The test statistic of a rank detector is a function of the ranks of the observations. Let the test statistic be denoted by $T(x_1, \ldots, x_{n^*})$, where n^* is the number of observations. A FSS rank detector for (1) performs the test:

$$T(x_1, \ldots, x_{n^*}) \begin{cases} > t^* \rightarrow \text{say } H_1 \\ = t^* \rightarrow \text{say } H_1 \text{ with prob. } \mu^* \\ \text{otherwise} \rightarrow \text{say } H_0, \end{cases} \quad (3)$$

where t^* is a threshold and μ^* is a randomization probability. Many rank test statistics $T(\cdot)$ have been studied in the literature. One of the most widely used rank tests is the Wilcoxon test. Let r_j be the rank of $|x_j|$, $1 \leq j \leq n^*$, among $|x_1|, |x_2|, \ldots, |x_{n^*}|$. Then, the Wilcoxon test statistic is defined as

$$T^W(x_1, \ldots, x_{n^*}) = \sum_{j=1}^{n^*} \text{sgn}\,(x_j) \cdot r_j \ , \quad (4)$$

where $\text{sgn}\,(x) = 1$ if $x \geq 0$ and $= 0$ if $x < 0$. For other types of rank test statistics, see [2]. In this paper, we will evaluate performances of M-stage detectors using test statistic (4) only. Other types of rank detectors are expected to have similar performances. The Wilcoxon test statistics can be written in the following form:

$$T^W(x_1, \ldots, x_{n^*}) = \sum_{j=1}^{n^*} \sum_{k=j}^{n^*} \text{sgn}\,(x_j + x_k) \quad (5)$$

which is more suitable for physical implementation. Note that the operation $\text{sgn}\,(x_j + x_k)$ is repeated $n^*(n^*+1)/2$ times. Therefore, the time required to calculate (5) (the ranking time) is proportional to $n^*(n^*+1)/2$.

The Wilcoxon detector has been shown to be very efficient (as measured by asymptotic relative efficiency) with respect to a linear detector [1]. However, the ranking time of the Wilcoxon detector grows quickly as the number of samples n^* increases. Schemes have been proposed to reduce the ranking time while the efficiency reduces slightly. See [3] as an example.

An M-stage detector is described in Section II. Its average sample size, power function, and asymptotic relative efficiencies (ARE) with respect to FSS detectors are given in Section III. Finite-sample and asymptotic performances of M-stage Wilcoxon detector are evaluated in Section IV, as well as discussions of some modifications.

II. M-Stage Rank Detector

Now, we describe an M-stage rank detector. In the i-th stage, $i = 1, \ldots, M$, we take n_i samples. With $n_0 = 0$, let the i-th group of samples $(x_{n_0 + \ldots + n_{i-1}+1}, \ldots, x_{n_0 + \ldots + n_i})$ be denoted by $\bar{x}_i$. An M-stage detector is described as follows:

stage i, $1 \le i < M$, test

$$T_i(\overline{x}_i) \begin{cases} > a_i \rightarrow H_1 \\ = a_i \rightarrow H_1 \text{ with prob. } \mu_{ai} \\ = b_i \rightarrow H_0 \text{ with prob. } \mu_{bi} \\ < b_i \rightarrow H_0 \\ \text{otherwise} \rightarrow \text{go to stage } i+1, \end{cases} \tag{6a}$$

and stage M, test

$$T_M(\overline{x}_i) \begin{cases} > a_M \rightarrow H_1 \\ = a_M \rightarrow H_1 \text{ with prob. } \mu_M \\ \text{otherwise} \rightarrow H_0 \end{cases} \tag{6b}$$

where $T_i(\overline{x}_i)$ is a rank test statistic which is based on the i-th group of observations $\overline{x}_i$. $T_i(\cdot)$ can be any rank statistics. For examples, $T_i(\cdot)$ is the Wilcoxon statistic for the i-th group of samples, corresponding to (4). Only this test statistic will be evaluated numerically. Note that only n_i samples need to be ranked in stage i. It will be seen in Section IV that (6) requires a smaller average ranking time than does detector (3) when both detectors have the same power function.

In order that the detector (6) is nonparametric, the thresholds a_1, ..., a_M, b_1, ..., b_{M-1} and the randomization probabilities μ_{a1}, ..., $\mu_{a(M-1)}$, μ_{b1}, ..., $\mu_{b(M-1)}$, μ_M must be chosen with only a symmetric pdf assumption of f(x). To achieve this, we define

$$\alpha_i = P(\text{say } H_1 \text{ in stage } i \mid \text{the test is in stage } i \text{ and } H_0 \text{ is true}) \tag{7}$$

for i = 1, ..., M and

$$\gamma_i = P(\text{say } H_0 \text{ in stage } i \mid \text{the test is in stage } i \text{ and } H_0 \text{ is true}) \tag{8}$$

for i = 1, ..., M-1. It follows that the false-alarm probability of (6) is given by

$$\alpha = \alpha_1 + \alpha_2 (1-\alpha_1-\gamma_1) + \dots + \alpha_M \prod_{j=1}^{M-1} (1-\alpha_j-\gamma_j). \tag{9}$$

If α is given, α_i and γ_i, i = 1, ..., M - 1, are chosen, then α_M can be calculated from (9). Once n_i, α_i and γ_i are fixed, the thresholds and randomization probabilities at stage i can be calculated using only the symmetric property of f(x), provided that $T_i(\cdot)$ is a nonparametric test statistic.

It is obvious that there are many possible combinations of n_i, α_i, and γ_i to choose from. The optimal choice depends on the criterion used. One criterion is to maximize the slope of the power function under H_0. This is suitable for small signal detection (local detector). A more physically meaningful criterion is to minimize the average sample size under H_0 and under H_1 with $\theta = \theta_1$, subject to a total number of observations $n_1 + \ldots + n_M = \tilde{n}$ or a detection probability of $\tilde{\beta}$ at $\theta = \theta_1$. However, this requires the knowledge of the pdf f(x). As an alternative, a nonoptimal design is possible with only the knowledge of symmetry of f(x) by studying asymptotic relative efficiency (ARE). The ARE of an M-stage detector with respect to its corresponding FSS detector is independent of f(x), provided that the Central Limit Theorem is obeyed by the test statistic. This ARE depends on the parameters α_i, γ_i, $k_i \triangleq n_i/n_1$, α, $\delta \triangleq \theta/\theta_1$, and detection probability $\tilde{\beta}$. If it is desired to minimize the average sample size under H_0 ($\delta = 0$) or under $\delta = 1$ or their weighted value with a specified α and $\tilde{\beta}$, the ARE (or weighted ARE) can be calculated for various values of α_i, γ_i, and k_i. The set of α_i, γ_i, and k_i which minimizes the ARE can be used. The average sample size saving for finite $\tilde{n}$ is less than that indicated by ARE (which is obtained by letting $\tilde{n} \to \infty$ and $\theta_1 \to 0$).

III. Power Function, Average Sample Size, and ARE

Calculations of the power function $\beta(\theta)$ and the average sample size $E(N|\theta)$ of (6) requires some conditional probabilities. These conditional probabilities are defined as follows:

$$\beta_i(\theta) = P_\theta \text{ (say } H_1 \text{ in stage } i \mid \text{the test reaches stage } i) \qquad (10)$$

for i = 1, ..., M, where $P_\theta(\cdot)$ is a probability given that the signal strength is θ, and

$$\gamma_i(\theta) = P_\theta \text{ (say } H_0 \text{ in stage } i \mid \text{the test reaches stage } i) \qquad (11)$$

for i = 1, ..., M-1. Note that under H_0, $\beta_i(0) = \alpha_i$ and $\gamma_i(0) = \gamma_i$. The average sample size is given by

$$E(N|\theta) = \sum_{i=1}^{M} \left(\sum_{j=1}^{i} n_j\right) P_\theta(\text{decision in stage } i) \qquad (12)$$

$$= n_1 + \sum_{i=2}^{M} n_i \prod_{j=1}^{i-1} (1-\beta_j(\theta) - \gamma_j(\theta)) \qquad (13)$$

and the power function is given by

$$\beta(\theta) = \beta_1(\theta) + \sum_{i=2}^{M} \beta_i(\theta) \prod_{j=1}^{i-1} (1-\beta_j(\theta) - \gamma_j(\theta)). \qquad (14)$$

The ARE is a large-sample, small-signal comparison between the average sample sizes of two detectors which achieve the same false-alarm and detection probabilities. Let $n^*(\alpha, \tilde{\beta}, \theta_1)$ be the sample size required by detector (3) to have a false-alarm proability of α and a detection probability of $\tilde{\beta}$ at θ_1. Likewise, let $E(N(\alpha, \tilde{\beta}, \theta_1)|\delta)$ be the average sample size of (6) with a false-alarm probability α and a detection probability of $\tilde{\beta}$ at θ_1. The parameter δ is the ratio θ/θ_1, where θ is the actual signal strength. Then, the ARE of M-stage detector (6) with respect to FSS detector (3) is defined as

$$\mathrm{ARE}_{MS,FSS}(\delta) = \lim \frac{n^*(\alpha, \tilde{\beta}, \theta_1)}{E(N(\alpha, \tilde{\beta}, \theta_1)|\delta)} , \tag{15}$$

where the limit is taken as $\theta_1 \to 0$, $n^* \to \infty$, and $\tilde{n} \to \infty$ such that $n_i/n_1 = k_i$ is constant, $i = 2, \ldots, M$. If $T(\cdot)$ in (3) and $T_i(\cdot)$ in (6) are the same type of test statistics and they obey the Central Limit Theorem, then expression for (15) can be derived via approximating the distribution function of $T(\cdot)$ by a normal distribution. The resulting ARE depends on the parameters k_i, α_i, and γ_i, in addition to α, $\tilde{\beta}$ and δ. To reduce the number of parameters, we let $k = k_2 = k_3 = \ldots = k_M$, $\alpha_2 = \alpha_3 = \ldots = \alpha_{M-1}$, and $\gamma_2 = \gamma_3 = \ldots = \gamma_{M-1}$. Because the performance of (6) is probably most sensitive to alterations of parameters in the first stage, we do not restrict α_1, γ_1 and n_1 to be the same as α_2, γ_2, and n_2, respectively. With a reduced set of parameters, the ARE of an M-stage detector with respect to a corresponding FSS detector can be derived and it is given by

$$\mathrm{ARE}_{MS,\ FSS}(\delta) = \frac{[\Phi^{-1}(\tilde{\beta}) + \Phi^{-1}(1-\alpha)]^2}{R^2[1+kQ(\delta)(1-Q(\delta))^{M-1}/(1-Q(\delta))]} , \tag{16}$$

where

$$Q(\delta) = \Phi(\Phi^{-1}(1-\alpha_1) - \delta R) - \Phi(\Phi^{-1}(\gamma_1) - \delta R), \tag{17}$$

and R is the solution of

$$\begin{aligned} \tilde{\beta} = {} & \Phi[R\sqrt{k} + \Phi^{-1}(\frac{\alpha_1}{\alpha_1+\gamma_1} + \frac{\alpha(\alpha_1+\gamma_1) - \alpha_1}{(\alpha_1+\gamma_1)(1-\alpha_1-\gamma_1)^{M-1}})]Q(1)Q_k^{M-2} \\ & + \Phi[R - \Phi^{-1}(1-\alpha_1)] + \Phi[R\sqrt{k} - \Phi^{-1}(1-\alpha_1)]Q(1)\frac{1-Q_k^{M-2}}{1-Q_k} , \end{aligned} \tag{18}$$

where

$$Q_k = \Phi[\Phi^{-1}(1-\alpha_1) - R\sqrt{k}] - \Phi[\Phi^{-1}(\gamma_1) - R\sqrt{k}]. \tag{19}$$

If it is desired to find the ARE of an M-stage detector with respect to a FSS detector D* with different type of test statistic , we can show that it is the product of (16) and the ARE of FSS detector (3) with respect to FSS detector D*, which can be found using Noether's results [4]. For example, the ARE of M-stage Wilcoxon detector with respect to the FSS linear detector is given by

$$ARE_{MS\text{-}W,LIN}(\delta) = ARE_{MS,FSS}(\delta) \cdot ARE_{W,LIN} \tag{20}$$

where $ARE_{W,LIN}$ is the ARE of the FSS Wilcoxon detector with respect to the FSS linear detector which is $12\sigma^2[\int f^2(x)dx]^2$, where σ^2 is the variance of X_i [1]. Therefore, we have

$$ARE_{MS\text{-}W,LIN}(\delta) = ARE_{MS,FSS}(\delta) \cdot 12\sigma^2[\int_{-\infty}^{\infty} f^2(x)dx]^2 \quad . \tag{21}$$

In the following section, performance of M-stage Wilcoxon detector will be evaluated.

IV. Performance of M-Stage Wilcoxon Detector

The ARE expression (16) can be evaluated given the parameter values. Some sets of parameters produce large ARE at $\delta = 0$ (under H_0), some others may produce large ARE at $\delta = 1$, and others may give large ARE at both $\delta = 0$ and $\delta = 1$. However, poor choices of parameters can result in ARE < 1 for $\delta\varepsilon[0,1]$. Generally, if ARE at $\delta = 0$ is made larger, the value at $\delta = 1$ would be smaller, or vice versa. It is not easy to draw any conclusions from (16) to (19) regarding good choices of parameters. To check whether a set of parameters is good or not, we simply calculate (16) numerically for various values of δ. One such example is given in Table I for α=0.001, $\tilde{\beta}$= 0.995, M= 6, α_1 = 0.0003, γ_2=0.8, and α_2=0.0008. Note that these choices are fairly good since they give high ARE at both $\delta = 0$ and $\delta = 1$. For k=1, ARE at $\delta = 0$ can be better than 1.9 and at $\delta = 1$ it is better than 1.5. However, it becomes smaller than unity for δ = 0.4 to 0.8. These results indicate that a well-designed M-stage detector can be (asymptotically) more efficient than its corresponding FSS detector at no-signal ($\delta = 0$) and large-signal ($\delta \geq 1$) situations. This efficiency comes at a cost of poor efficiency for δ near 0.5. However, δ near 0.5 is not a region where a detector is designed to operate since the detection probability is too small in this region. Rather, a detector is usually designed to operate at $\delta = 0$ and $\delta \geq 1$.

Let us evaluate finite-sample performance of an M-stage Wilcoxon detector. At stage i, the test statistic $T_i(\cdot)$ is the Wilcoxon statistic (4) (n^* replaced with n_i). The conditional probabilities $\gamma_i(\theta)$ and $\beta_i(\theta)$ are calculated numerically according to methods given by Klotz [5], which requires n_i successive integrations. Two numerical results are plotted in Figures 1 and 2 for α= 0.02 and 0.01, respectively, where the noise pdf f(x) is a unit-variance normal noise. Power functions of FSS linear detectors are also plotted for comparison. Compared to the FSS linear detector (optimal FSS detector for normal noise) with comparable power function, the M-stage Wilcoxon detector shows significant saving in average sample size for θ near 0 and for large θ. The saving is confirmed by the ARE, in Table II (a) and (b), of M-stage Wilcoxon detector with respect to FSS linear detector (calculated using (16)-(19) and (21)) with the same sets of parameter values as those in Figures 1 and 2. The saving is more prominent if it is realized that the FSS linear detector is a parametric detector while the M-stage Wilcoxon detector is a nonparametric one. Under H_0, the finite-sample efficiency is close to the ARE. Numerical results show that the ARE at $\delta=1$ is higher if $\tilde{\beta}$ is closer to unity.

Now, let us compare the average ranking time of an M-stage Wilcoxon detector to the ranking time of a FSS Wilcoxon detector with comparable power function. It was mentioned earlier that a FSS Wilcoxon detector with n^* samples requires a ranking time proportional to $n^*(n^*+1)/2$. But an M-stage Wilcoxon detector requires an average ranking time proportional to

$$\sum_{i=1}^{M}\left(\sum_{j=0}^{i-1}\frac{n_j(n_j+1)}{2}\right)P_\theta\text{ (test terminates in stage } i\text{)}, \tag{22}$$

where $n_0 \equiv 0$. If $n_1 = \ldots = n_M$, (22) reduces to $(n_1+1)E(N|\theta)/2$. One example is shown (together with percentage saving) in Table IV for the same parameter values as those of Figure 2. It can be seen that the ranking time has been reduced significantly for all values of θ. This emphasizes another advantage of M-stage rank detectors. Other cases show similar savings.

If we wish to improve the performance of an M-stage rank detector without increasing the ranking time, one possible modification is to add the values of the test statistics of previous stage(s) to the test statistic of present stage. However, now the thresholds design and power function calculation can become very involved. The conditional distribution function of the modified test statistic at stage i (given that no decision has been made in previous stages) can be obtained by convoluting the distribution of $T_i(\cdot)$ with the conditional distribution function of the modified test statistic at stage i-1 (given that no decision prior to stage i-1 and no decision in stage i). In principle, the conditional distribution function can be seccessively calculated for i = 1, 2, ..., M. Another modification is to relax the conditions $k_2=\ldots=k_M$, $\gamma_2=\ldots=\gamma_{M-1}$, and $\alpha_2=\ldots=\alpha_{M-1}$. The ARE can be derived and the optimal set of parameters can be found by an optimization procedure.

V. Conclusion

We have studied in detail finite-sample performances and asymptotic relative efficiencies of M-stage Wilcoxon detector. Besides reducing the average ranking time, significant average sample size reductions (compared to FSS detector) under no-signal and large-signal cases have been observed for proper choices of parameters. The average sample size reductions (in normal noise) still hold when an M-stage Wilcoxon detector is compared to a FSS linear detector with comparable power function, despite the fact that an M-stage Wilcoxon detector is nonparametric while a FSS linear detector is parametric.

VI. Acknowledgement

This research was supported by the Engineering Experiment Station, Auburn University.

References

1. J. B. Thomas, "Nonparametric detection", Proc. IEEE, Vol. 58, pp. 623-631, May 1970.
2. E. L. Lehmann, Nonparametrics: Statistical Methods Based on Ranks. San Francisco: Holden-Day, Inc., 1975.
3. E. A. Feustal and L. D. Davisson, "The asymptotic relative efficiency of mixed statistical tests", IEEE Trans. on Inform. Theory, Vol. IT-13, pp. 247-255, April 1967.
4. G. E. Noether, "On a theorem of Pitman", Ann. Math. Stat., Vol. 26, pp. 64-68, 1955.
5. J. H. Klotz, "Small sample power and efficiency for the one-sample Wilcoxon and normal scores tests", Ann. Math. Stat., Vol. 34, pp. 624-632, 1963.

Table I. $ARE_{MS-FSS}(\delta)$ for $\alpha=0.001$, $\tilde{\beta}=0.995$, $M=6$, $\alpha_1=0.0003$, $\gamma_2=0.8$.

(a) $\alpha_2=0.0008$, $\gamma_1=0.7$

δ	k=0.9	k=1.0
0.0	1.791	1.949
0.2	1.267	1.352
0.4	0.842	0.878
0.5	0.758	0.765
0.6	0.772	0.743
0.8	1.085	0.969
1.0	1.623	1.511

(b) $\alpha_2=0.0008$, $\gamma_1=0.4$

δ	k=0.9	k=1.0
0.0	1.540	1.602
0.2	1.117	1.160
0.4	0.801	0.828
0.5	0.735	0.748
0.6	0.745	0.740
0.8	1.020	0.966
1.0	1.576	1.506

Table II. ARE of M-stage Wilcoxon detector with respect to FSS linear detector $ARE_{MS-W,LIN}(\delta)$ for normal noise.

(a) $\alpha=0.02$, $\tilde{\beta}=0.95$, $M=5$, $\alpha_1=0.012$, $\alpha_2=0.015$, $\gamma_1=\gamma_2=0.7$

δ	k=0.8	0.9	1.0
0.0	1.57	1.60	1.62
0.2	1.27	1.28	1.29
0.4	1.06	1.06	1.05
0.5	1.02	1.01	0.99
0.6	1.02	1.00	0.98
0.8	1.16	1.13	1.10
1.0	1.43	1.40	1.37

(b) $\alpha=0.01$, $\tilde{\beta}=0.99$, $M=6$, $\alpha_1=\alpha_2=0.007$, $\gamma_1=\gamma_2=0.7$

δ	k=0.8	0.9	1.0
0.0	1.61	1.63	1.65
0.2	1.19	1.20	1.19
0.4	0.92	0.91	0.89
0.5	0.90	0.88	0.85
0.6	0.95	0.92	0.89
0.8	1.28	1.23	1.19
1.0	1.68	1.67	1.65

Table III. Ranking time and average ranking time of FSS Wilcoxon and M-stage Wilcoxon detectors with comparable power functions. Unit-variance normal noise, $M=6$, $\alpha_1=\alpha_2=0.007$, $\gamma_1=\gamma_2=0.7$, $n_i=8$ (all i), $n^*=18$, $\alpha=0.01$.

θ	(Average) Ranking Time FSS Wilcoxon $n^*=18$	M-stage Wilcoxon	% saving
0.0	171	51.4	69.9
0.2	171	70.1	59.0
0.4	171	98.0	42.7
0.6	171	118.1	30.9
0.8	171	112.3	34.3
1.0	171	89.7	47.5
1.2	171	67.6	60.5

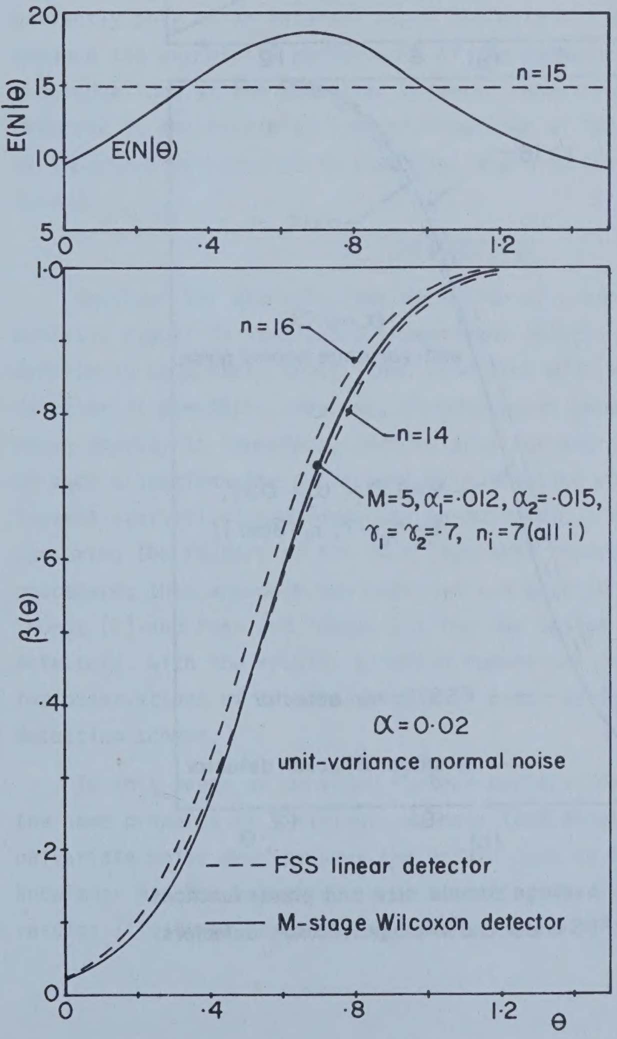


Figure 1 Average sample size and power functions of FSS linear and M-stage Wilcoxon detectors.

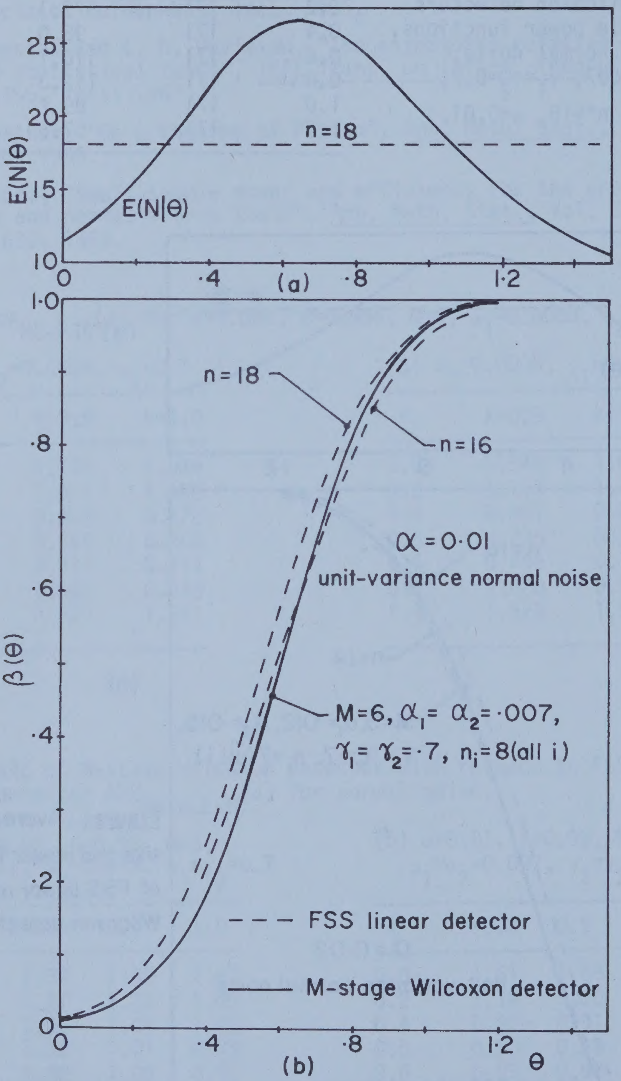


Figure 2 Average sample size and power functions of FSS linear and M-stage Wilcoxon detectors.

ON THE USE OF LIMITING IN CONJUNCTION WITH THE DETECTION OF SIGNALS IN PARTIALLY CONTAMINATED NOISE

D.R. HALVERSON
Department of Electrical Engineering
Texas A&M University
College Station, Texas 77843

ABSTRACT

We consider the discrete time detection of a known time varying deterministic signal in white noise, where the univariate noise density is known perfectly only on an interval about the origin. We present a method to enhance the asymptotic performance of the detector by exploiting this knowledge, and at the same time preserve robustness properties of the detector to the remaining inexact knowledge of the univariate noise density. We then provide examples to show that improved performance is indeed obtained.

I. INTRODUCTION

Consider the discrete time detection of a known time varying deterministic signal in stationary independent noise. If the univariate noise density is completely known, then selection of a Neyman-Pearson optimal detector is possible. However, in many cases knowledge of the univariate noise density is imperfect, particularly for extreme values of its argument. In such situations the employment of a detector which is robust to the inexact statistical knowledge may be desirable. Huber observes in [1] that censoring the factors of the test statistic imparts robustness into the procedure; this approach has been applied by, for example, Kassam and Thomas [2] and Poor and Thomas [3] for the design of asymptotically robust detectors, with the results pointing toward the introduction of limiting for observations of large magnitude in order to impart robustness into the detection scheme.

In this paper we consider further applications of detectors possessing the same property of limiting. We note that often more is known about the univariate noise density near the origin than on the tails, and such knowledge has the potential to be exploited for improved performance. Our results illustrate how such improved performance can be obtained and still preserve the robustness of the detector to the remaining inexact knowledge of the univariate noise density through the employment of limiting.

II. DEVELOPMENT

Suppose we denote the discrete time noise process by $\{N_i; i = 1, 2, \ldots\}$, with $f(\cdot)$ denoting the associated univariate noise density, and let the time varying deterministic signal be denoted by $\{s_i; i = 1, 2, \ldots\}$. The detection problem then reduces to a choice between

$$H_0 : Y_i = N_i \; ; \; i = 1, 2, \ldots, n$$

$$H_1 : Y_i = N_i + \theta s_i \; ; \; i = 1, 2, \ldots, n \quad ,$$

where we observe realizations y_i of the random variables Y_i. As in [2] and [3], we assume the noise density $f(\cdot)$ is a member of the Huber-Tukey mixture class, i.e. $f(\cdot)$ can be expressed as $f(\cdot) = (1 - \varepsilon)\, f_0(\cdot) + \varepsilon \cdot h(\cdot)$, where ε is a fixed known number between zero and unity, $f_0(\cdot)$ is a known nominal density which is twice continuously differentiable and symmetric with support $(-\infty, \infty)$, and $h(\cdot)$ is an unknown symmetric "contaminating" density. In addition, however, we assume that $f(x)$ is known exactly for each x in an interval $[-a, a]$, i.e. $h(x) = f_0(x)$ for all x in $[-a, a]$; noise with such a univariate density $f(\cdot)$ might thus be called "partially contaminated". Motivated by the results of [2] and [3], as well as the form of the locally optimal detector given by [4], we employ the class of detectors where a test statistic of form

$T_g(y) = \sum_{i=1}^{n} s_i \, g(y_i)$ is compared to a threshold, with the nonlinearity

$$g(\cdot) \text{ given by } g(x) = \begin{cases} -f_0'(x)/f_0(x) & \text{if } |x| \leq k \\ -\operatorname{sgn}(x)\, f_0'(k)/f_0(k) & \text{otherwise} \end{cases} , \text{ where } k \geq a.$$

We thus see that specifying k effectively specifies the detector.

In order to analyze and compare the performance of the detector, and hence determine k for best performance, an asymptotic fidelity criterion is employed. Because of its relationship to the Pitman-Noether theorem [5], the criterion of Asymptotic Relative Efficiency (ARE) is particularly useful; the associated efficacy functional η provides a convenient measure of performance, where η is defined by

$$\eta(k,h) \triangleq \frac{\left[\int_{-\infty}^{\infty} g'(x)\ f(x)\ dx\right]^2}{\int_{-\infty}^{\infty} g^2(x)\ f(x)\ dx},$$

with the indicated dependence of η on k and $h(\cdot)$. Note that the ARE of two detectors in the class considered is given as the ratio of efficacies of the above form provided the hypothesis of the Pitman-Noether theorem [5] is satisfied with $\theta = c/\sqrt{n}$ for some $c > 0$, and provided that

$$\frac{\partial}{\partial\theta} \int_{-\infty}^{\infty} g(x+\theta s_i)\ f(x)\ dx \Bigg|_{\theta=0} = \int_{-\infty}^{\infty} \frac{\partial}{\partial\theta} g(x+\theta s_i)\ f(x) \Bigg|_{\theta=0} dx$$

for i = 1, 2, ... n. In the sequel we will employ the aforementioned definition of efficacy to measure performance, with the understanding of the relevance of efficacy to the ARE under the regularity conditions alluded to above.

In order to preserve the robustness property of the detector, we then choose k according to $\sup_k \inf_h \eta(k,h)$; such a choice of k will be called an optimal choice. The following result provides insight into how to obtain an optimal choice of k:

Theorem: Suppose $g(\cdot)$ is nondecreasing with positive left hand derivative at k. Then a necessary condition for an optimal choice of $k > a$ is given by

$$\begin{aligned}0 = {} & 4(1-\varepsilon)\ f_0(k) \int_{-a}^{a} g(x)^2 f_0(x)dx + 8(1-\varepsilon)^2\ f_0(k) \int_{a}^{k} g(x)^2 f_0(x)dx \\ & +8(1-\varepsilon)^2\ f_0(k)g(k)^2 \int_{k}^{\infty} f_0(x)dx + 8\varepsilon(1-\varepsilon)\ f_0(k)g(k)^2 \int_{a}^{\infty} f_0(x)dx \\ & -8\varepsilon(1-\varepsilon)g(k) \int_{a}^{k} g'(x)f_0(x)dx \int_{a}^{k} f_0(x)dx - 4g(k) \int_{-a}^{a} g'(x)f_0(x)dx \int_{k}^{\infty} f_0(x)dx \\ & -8(1-\varepsilon)g(k) \int_{a}^{k} g'(x)f_0(x)dx \int_{k}^{\infty} f_0(x)dx - 4\varepsilon g(k) \int_{-a}^{a} g'(x)f_0(x)dx \int_{a}^{k} f_0(x)dx.\end{aligned}$$

Proof: Evaluating $\eta(k,h)$ we have

$$\eta(k,h) = \frac{\left[\int_{-a}^{a} g'(x)f_0(x)\,dx + 2(1-\varepsilon)\int_{a}^{k} g'(x)f_0(x)\,dx + 2\varepsilon\int_{a}^{k} g'(x)h(x)\,dx\right]^2}{\left[\int_{-a}^{a} g(x)^2 f_0(x)\,dx + 2(1-\varepsilon)\int_{a}^{k} g(x)^2 f_0(x)\,dx + 2\varepsilon\int_{a}^{k} g(x)^2 h(x)\,dx\right.}.$$

$$\left. + 2(1-\varepsilon)g(k)^2\int_{k}^{\infty} f_0(x)\,dx + 2\varepsilon\, g(k)^2\int_{k}^{\infty} h(x)\,dx\right]$$

Note that $\int_{k}^{\infty} h(x)\,dx = \int_{a}^{\infty} f_0(x)\,dx - \int_{a}^{k} h(x)\,dx \triangleq b - \int_{a}^{k} h(x)\,dx,$

and thus

$$\int_{a}^{k} g(x)^2\, h(x)\,dx + g(k)^2\int_{k}^{\infty} h(x)\,dx = \int_{a}^{k} (g(x)^2 - g(k)^2)\, h(x)\,dx + bg(k)^2$$

$$\leq bg(k)^2.$$

We then observe that the denominator of $\eta(k,h)$ is then maximized over $h(\cdot)$ by choosing $h(x) = 0$ for $x \epsilon (a,k)$. Noting that such a choice simultaneously minimizes the numerator of $\eta(k,h)$, we therefore conclude

$$\inf_{h} \eta(k,h) = \frac{\left[\int_{-a}^{a} g'(x)f_0(x)\,dx + 2(1-\varepsilon)\int_{a}^{k} g'(x)\, f_0(x)\,dx\right]^2}{\left[\int_{-a}^{a} g(x)^2\, f_0(x)\,dx + 2(1-\varepsilon)\int_{a}^{k} g(x)^2 f_0(x)\,dx\right.}.$$

$$\left. + 2(1-\varepsilon)\, g(k)^2\int_{k}^{\infty} f_0(x)\,dx + 2\varepsilon\, g(k)^2\int_{a}^{\infty} f_0(x)\,dx\right]$$

The required necessity condition then follows by differentiating the above with respect to k. QED

We now consider the case where the noise is nominally zero mean unit variance Gaussian, and compare the "worst case" efficacy $\eta_{2,w}$ of the detector of [2] (i.e. the minimal efficacy over those $h(\cdot)$ satisfying our constraints as well as those of [2]) with the "worst case" efficacy $\eta_{1,w}$ of our detector with k chosen via the Theorem (where we set k = a if no solution exists). The results, which depend on the choice of a and ε, are tabulated below:

Table: Values of $\eta_{1,w}/\eta_{2,w}$

	ε=.1	ε=.2	ε=.3	ε=.4	ε=.5	ε=.6	ε=.7	ε=.8	ε=.9
a=2.00	1.07	1.13	1.18	1.23	1.28	1.33	1.38	1.43	1.49
a=1.00	1.00	1.03	1.08	1.12	1.17	1.21	1.26	1.31	1.36
a= .50	1.00	1.09	1.11	1.05	1.02	1.06	1.11	1.15	1.20
a= .20	1.00	1.21	1.45	1.66	1.77	1.68	1.30	1.02	1.06
a= .10	1.00	1.25	1.61	2.03	2.48	2.86	2.84	2.03	1.01
a= .05	1.00	1.28	1.70	2.26	3.02	4.00	4.96	4.89	2.26

Note that, somewhat surprisingly, the most dramatic improvement in performance occurs when much uncertainty is present in our knowledge of the univariate noise density (large ε and small a). Finally, we observe that for one parameter fixed, the efficacy ratio is not necessarily monotone in the other parameter.

III. CONCLUSION

We have introduced an approach toward improving the asymptotic performance of a detector by exploiting available knowledge of the univariate noise density near the origin, while at the same time retaining the robustness of the detector to inexact knowledge of the tails of the noise density. We have shown by way of example that such improved "worst case" performance does indeed occur.

ACKNOWLEDGEMENT

This research was supported by the Air Force Office of Scientific Research under Grant AFOSR-82-0033. The author would also like to acknowledge the assistance of Mr. Michael Thompson in preparing the Table.

REFERENCES

1. P.J. Huber, Robust Statistics, pp. 264-273, New York: Wiley, 1981.

2. S.A. Kassam and J.B. Thomas, "Asymptotically robust detection of a known signal in contaminated non-Gaussian noise", IEEE Trans. Inform. Theory, vol. IT-22, pp. 22-26, Jan. 1976.

3. H.V. Poor and J.B. Thomas, "Robust detection of stochastic signals", Proc. Fifteenth Annual Allerton Conf. on Communication, Control, and Computing, Monticello, Ill. Sept. 28-30, 1977.

4. J.H. Miller and J.B. Thomas, "Detectors for discrete-time signals in non-Gaussian noise", IEEE Trans. Inform. Theory, vol. IT-18, pp. 241-250, March 1972.

5. G.E. Noether, "On a theorem of Pitman", Ann. Math. Stat., vol. 26, pp. 64-68, 1955.

ON NARROWBAND IMPULSIVE NOISE*

KENNETH S. VASTOLA
Department of Electrical Engineering
and Computer Science
Princeton University
Princeton, N.J. 08544

ABSTRACT

The Middleton Class A impulsive noise model [9-12] is examined. It is shown that this noise model (which is known to fit closely a variety of non-Gaussian noises) can itself be closely approximated by a computationally much simpler noise model. It is then shown via numerical examples that for the problem of locally optimum detection the simplest form of this approximation yields nearly optimal (asymptotic) performance. Finally, a useful relationship between the Class A model and the ε-mixture model is developed.

I. INTRODUCTION

For many communication problems the usual Gaussian noise assumption is inadequate. Often this is due to the occurrence of low probability, high amplitude "spikes". This impulsive component of the interference has been found to be significant in many problems especially those involving atmospheric noise where lightning discharges in the vicinity of the receiver can cause such spikes [1-3] and in problems such as sonar and submarine communication where underwater acoustical noises include impulses due to noisy aquatic animals such as snapping shrimp [4] and, in arctic regions, due to ice cracking [5]. In addition to these natural impulsive noise sources, there is a great variety of man-made impulsive noise sources such as automobile ignitions, neon lights, and other electronic devices [6-11].

Various attempts have been made to develop models of impulsive noises. These models can be divided into two groups: those which are empirically motivated and those which are physically motivated. Empirical models [13,14] are those developed to fit collected data often with little regard for the underlying physical mechanisms. Physical models [7-12,15-17], on the other hand, attempt to model these mechanisms directly.

Among the physical models of impulsive noise some of most general are those developed by Middleton [7-12]. Middleton divides impulsive noise into two classes, A and B. (There has also been consideration of a Class C which contains noises which are sums of Class A and Class B components [9].) Class B noises are broadband, i.e. those with spectra broader than the passband of the receiver. Class A noises are narrowband, i.e. have spectra comparable to or narrower than the receiver passband.

In this paper we examine the Middleton Class A noise model. We show that in a wide variety of cases the first-order noise probability density function (PDF) can be closely approximated by PDF's having considerably simpler form. Further, in Section IV, a relationship is developed between the Class A noise model and the ε-mixture model. The ε-mixture model is a highly tractable empirical model, and through this relationship the advantages of the (physical) Class A model can be carried over to the mixture model (e.g. physical motivation and direct calculation of basic parameters).

II. THE CLASS A NOISE MODEL

Middleton [7-12] assumes the noise has the form $X(t) + N(t)$ where $N(t)$ is a Gaussian background component and

$$X(t) = \sum_j U_j(t, \vartheta) \tag{1}$$

---

*This work was supported by the Office of the Naval Research under Contract N00014-81-K-0146

where U_j denotes the j^{th} received waveform from an interference source and ϑ is a random parameter. He then assumes that the waveform receptions are Poisson distributed in time and shows that the normalized (to unit variance) noise density $f(x)$ can be canonically approximated by

$$f(x) = \sum_{m=0}^{\infty} K_m f(x;\sigma_m^2) \tag{2}$$

where $f(x;\sigma^2)$ is the zero-mean Gaussian PDF with variance σ^2. The variance σ_m^2 of the m^{th} density is given by

$$\sigma_m^2 = \frac{m/A+\Gamma}{1+\Gamma} \tag{3}$$

and the coefficient K_m is given by

$$K_m = e^{-A}\,\frac{A^m}{m!} \tag{4}$$

where A and Γ are the two basic parameters of the model. The first basic parameter, A, is called the "impulsive index" and is defined by $A = vT$ where v is the rate of the homogeneous Poisson process which governs the generation of the interfering waveforms U_j and T is the mean duration of a typical interfering signal. The other basic parameter, Γ, is given by the ratio of the power in the Gaussian portion of the interference to the power in the Poisson component. The corresponding envelope distribution is given by

$$P(E>E_0) = \sum_{m=0}^{\infty} K_m\, e^{-E_0^2/\sigma_m^2} \tag{5}$$

Middleton has shown that by adjusting the parameters A and Γ the density f given in (2) can be made to fit a great variety of impulsive noise densities [9-12]. Also the parameters A and Γ are physically motivated and can be directly estimated (see [12,9]).

Unfortunately the model (2) is cumbersome. For example, in [11], Spaulding and Middleton exhibit the optimal nonlinearity for detection (i.e. the likelihood ratio $f(x-s_1)/f(x-s_0)$) and point out that this detector structure is likely to be computationally burdensome and uneconomical. Thus we would like to develop detector nonlinearities having simpler structure but which retain the desirable properties of the one given by Spaulding and Middleton.

III. APPROXIMATING THE CLASS A MODEL

For each $M \geq 1$ we define the PDF f_M to be the (normalized) M-term truncation of the Class A noise PDF given in (2), i.e.

$$f_M(x) = \frac{\sum_{m=0}^{M-1} K_m f(x;\sigma_m^2)}{\sum_{m=0}^{M-1} K_m} \tag{6}$$

Since the K_m's are positive and $\sum_{m=0}^{\infty} K_m = 1$, we have that f_M (given in (6)) converges pointwise to f (given in (2)) as $M \to \infty$. (In fact, it converges uniformly). Our goal in this section is to show that f_M is actually a very good approximation to f for small values of M.

That the Class A noise PDF given in (2) is a good model for narrowband impulsive noise has been demonstrated [9-12] by showing that (for appropriate choice of parameters A and Γ) the envelope distribution (5) closely fits the measured envelope distribution of various Class-A-type noises (e.g. interference from powerlines or machinery). In Figures 1-4 we have plotted the M- term (normalized) truncation of the envelope distribution (5) for M=1,2,3,... We see that in each case only two or three terms are necessary in order that the truncated envelope distribution be indistinguishable from the true one. The parameters (A=.35,Γ=.0005) for Figure 1 are used by Middleton [9-12] to fit "interference (probably) from nearby powerline, produced by some kind of equipment fed by line" [9]. Figure 2's parameters (A=.0001,Γ=50) fit data from ore-crushing machinery. The parameters of Figure 3 ($A=.1, \Gamma=.001$) and of Figure 4 ($A=.1, \Gamma=.1$) are from a range of typical values [9-12].

Figures 1-4 give rise to the hope that a detector designed to be optimal assuming the noise PDF is f_M (with M small) would perform well when the actual PDF of the noise is f. We consider here the case of "locally optimum" detection (i.e. small signal, large time-bandwidth product). The details of locally optimum detection have been presented in many places [24-26,11]. We will only state the needed results.

Under mild regularity conditions the (asymptotic) performance (or processing gain) achievable using a given detector nonlinearity g(x) when the i.i.d. noise has PDF h(x) is given by the efficacy function

$$L(g,h) = \frac{\left[\int_{-\infty}^{\infty} g(x)h'(x)dx\right]^2}{\int_{-\infty}^{\infty} g^2(x)h(x)dx} \tag{7}$$

For a given noise PDF h(x) the locally optimum detector nonlinearity g*(x) is the solution to

$$L^*(h) = \max_g L(g,h)$$

and is given by

$$g^*(x) = \frac{-h'(x)}{h(x)} \tag{8}$$

If we let $h=f_M$ in (8) we obtain the locally optimum detector nonlinearity $g_M^*(x)$ for f_M. Since f_M is symmetric, $g_M^*(x)$ is antisymmetric (i.e. $g_M^*(-x)=-g_M^*(x)$). In Figures 5-8 $g_M^*(x)$ is plotted for the examples of Figures 1-4, respectively, for $x>0$ and M=2,3,4,... (for M=1, $g_M^*(x)$ is a straight line with slope $1/\sigma_0^2=(1+\Gamma)/\Gamma$). We see that in each case $g_2^*(x)$ closely approximates the locally optimum nonlinearity for f.

Returning to the general case, we have that for any PDF h(x) the performance of its locally optimum nonlinearity g*(x) is given by

$$L^*(h) = L(g^*,h) = \int_{-\infty}^{\infty} \left[\frac{h'(x)}{h(x)}\right]^2 h(x)dx \tag{9}$$

L*(h) is also known as Fisher's Information for h. An interesting (and well known) fact about the function L*(h) is that it is minimized by the Gaussian PDF. In fact, the locally optimum detector nonlinearity for Gaussian noise is the linear detector ($g(x)=x/\sigma^2$) which has performance equal to one for all noise PDF's.

Since the Class A noise PDF f in (2) is highly non-Gaussian we will often have $L^*(f)>>1$. We have seen that for very small values of M, f_M and $g_M^*(x)$ closely approximate f and its locally optimum nonlinearity $g_f^*(x)$. So there is reason to believe that $L(g_M^*,f)$ should be close to L*(f). Table 1 bears this out. For each of the examples which we have been considering (which cover a fairly wide range of realistic values of the parameters) we see that the processing gain achievable using g_2^* is extremely close to that achievable using g_f^*.

IV. THE ε-MIXTURE MODEL

In this section we consider the relationship between the ε-mixture model (defined below) and Middleton's Class A model. In the Introduction we mentioned two broad classes of models for impulsive noise: physical models (such as the Middleton models) and empirical models. One commonly used empirical model is the ε-mixture (or ε-contaminated) model in which the first-order noise PDF has the form

$$f_\varepsilon(x) = (1-\varepsilon)f_0(x) + \varepsilon f_1(x) \tag{10}$$

where $\varepsilon \in [0,1]$ and f_0 and f_1 are PDF's. The PDF f_0 is usually taken to be a Gaussian PDF representing background (i.e. nonimpulsive) noise such as receiver front-end noise. Among the choices for the "contaminating" PDF f_1 are various "heavy-tailed" densities such as the Laplacian or double exponential [18,19]. Others have chosen f_1 to be also Gaussian, but with variance $\sigma_{f_1}^2$ taken to be many times the variance of f_0, $\sigma_{f_1}^2$. The ratio $\gamma^2 \triangleq \sigma_{f_1}^2 / \sigma_{f_0}^2$ has generally been taken to be between 1 and 100 [20-22]. As with other empirical models the disadvantage of the mixture model is that the parameters (ε and γ^2) are not directly related to the underlying physical situation and hence are difficult to determine. The primary advantage of the mixture model is its analytic simplicity.

We saw in the previous section that for a wide range of values of the parameters, A and Γ, of Middleton's Class A noise model the (normalized) sum of the first two terms of the first-order noise PDF give a sufficiently accurate approximation to the full PDF given in (2). In these cases f_M is just a "Gauss-Gauss" mixture as in (10) with $f_0(x)=f(x;\sigma_0^2)$, $f_1(x)=f(x;\sigma_1^2)$,

$$\gamma^2 = \frac{\sigma_1^2}{\sigma_0^2} = 1 + \frac{1}{A\Gamma} \tag{11}$$

and

$$\varepsilon = \frac{K_1}{K_0 + K_1} = \frac{A}{1 + A} \tag{12}$$

Thus we see that, when a Gauss-Gauss mixture is used to model narrowband non-Gaussian noise, it may be possible to obtain ε and γ^2 from (11), (12) and the techniques already developed for determining A and Γ [12,9].

Another interesting consequence of the relations (11) and (12) is that (at least, for Class-A-type noises) reasonable values for γ^2 seem to be between 100 and 10^4. For example, if A=.1 and Γ=.001 as in Figures 3 and 7 then γ^2=10,001, and if A=.35 and Γ=.0005 as in Figures 1 and 5 then γ^2=5715. These values are as much as two orders of magnitude greater than those previously used for γ^2 [20-22]. On the other hand the commonly used values of ε, say between .01 and .25, correspond approximately to values of A between .01 and 1/3.

V. SUMMARY AND CONCLUSIONS

In this paper we have shown that the Middleton Class A noise model can often be closely approximated by the (normalized) sum of just the first few terms. In fact, in many cases, two terms are sufficient. This was especially clear when we looked at the efficacy of the detector nonlinearity which is locally optimum for two terms of the Class A model and found it comparable to the efficacy of the full locally optimum detector (see Table 1).

We have also shown how the above approximation leads to a relationship between the Class A model and the ε-mixture model. This yields a way of estimating the parameters of the mixture model directly from the physically motivated parameters of the Class A model.

It should also be noted that one of the clear advantages of the Middleton Class A noise model is that, since its parameters can be estimated directly from the data, any detector based on this model could easily be implemented adaptively. Such a situation would require real-time computation of the detector nonlinearity. Here the above approximation would result in substantial computational savings. Furthermore, in such a situation it would be possible to determine M (the truncation parameter) in an adaptive fashion as well. That is, first determine A and Γ and then choose the number of terms necessary to achieve the desired degree of approximation. In fact, this approach (of determining M adaptively) could actually be used to obtain optimal (not just nearly optimal) performance in some situations. Optimal performance could be achieved using the philosophy of the Schwartz detector (see [23] for details).

Finally we note that, while we have carried out our analysis only for the problem of locally optimum detection, the closeness of the envelope distribution approximations (Figures 5-8) encourage us to believe that similar computational savings might be realized by applying the approach of this paper to other non-Gaussian signal processing problems.

ACKNOWLEDGEMENT

The author wishes to thank S.V. Czarnecki for helpful discussions regarding the numerical results of this paper.

REFERENCES

1. A.D. Watt and E.L. Maxwell, "Characteristics of Atmospheric Noise from 1 to 100 KC," *Proc. IRE*, Vol. 45, pp. 787-794, June 1957.
2. A.D. Watt and E.L. Maxwell, "Measured Statistical Characteristics of VLF Atmospheric Radio Noise," *Proc. IRE*, Vol. 45, pp. 55-62, January 1957.
3. A.D. Watt, *V.L.F. Radio Engineering* Pergamon Press, New York, 1967.
4. D.W. Lytle, "Characteristic Problems of Underwater Communications," *Convention Record ICC*, 1973.
5. A.R. Milne and J.H. Ganton, "Ambient Noise Under Arctic Sea Ice," *Journal Acoustical Soc. Amer.*, Vol. 36, No. 5, pp. 855-863, May 1964.
6. E.N. Skomal, "Distribution and Frequency Dependence of Unintentinally Generated Man-Made VHF/UHF Noise in Metropolitan Areas. Part II - Theory," *IEEE Trans. on Electromagnetic Compatibility*, Vol. EMC-7, pp. 420-427, December 1965.
7. D. Middleton, "Statistical-Physical Models of Urban Radio-Noise Environments, Part I: Foundations, *IEEE Trans. Electromagn. Compat.*, Vol. EMC-14, pp. 38-56, May 1972.
8. D. Middleton, "Man-Made Noise in Urban Environments and Transportation Systems," *IEEE Trans. Commun.*, Vol. COM-21, pp. 1232-1241, Nov. 1973.
9. D. Middleton, "Statistical-Physical Models of Electromagnetic Interference," *IEEE Trans. Electomagn. Compat,*. Vol. EMC-19, pp. 106-127, Aug. 1977.
10. D. Middleton, "Canonical Non-Gaussian Noise Models: Their Implications for Measurement and for Prediction of Receiver Performance," *IEEE Trans. Electromagn. Compat.*, Vol. EMC-21, pp. 209-220, Aug. 1979.
11. A.D. Spaulding and D. Middleton, "Optimum Reception in an Impulsive Interference Environment - Part I: Coherent Detection," *IEEE Trans. Comm.*, Vol. COM-25, No. 9, pp. 910-923, September 1977.
12. D. Middleton, "Procedures for Determining the Parameters of the First-Order Canonical Models of Class A and Class B Electromagnetic Interference," *IEEE Trans. Electromagn. Compat.*, Vol. EMC-21, pp. 190-208, Aug. 1979.
13. W.Q. Crichlow, C.J. Roubique, A.D. Spaulding, and W.M. Beery, "Determination of the Amplitude-Probability Distribution of Atmospheric Radio Noise from Statistical Moments," *Journal of Research NBS*, Vol. 65D (Radio Prop.), No. 1, pp. 49-56, January 1960.
14. F. Horner and J. Harwood, "An Investigation of Atmospheric Radio Noise at Very Low Frequencies," *Proc. Inst. Elec. Engrs.*, No. 103, Pt. B, pp. 743-751, November 1956.
15. K. Furutsu and T. Ishida, "On the Theory of Amplitude Distribution of Impulsive Random Noise," *J. of Applied Physics*, Vol. 32, No. 7, pp. 1206-1221, July 1961.
16. P. Beckmann, "Amplitude-Probability Distribution of Atmospheric Radio Noise," *Journal of Research NBS*, Vol. 68D (Radio Science), No. 6, pp. 723-736, June 1964.
17. A.A. Giordano and F. Haber, "Modeling of Atmospheric Noise, *Radio Science*, Vol. 7, No. 11, pp. 1101-1023, November 1972.
18. J.H. Miller and J.B. Thomas, "The Detection of Signals in Impulsive Noise Modeled as a Mixture Process," *IEEE Trans. Communication*, Vol. COM-24, pp. 559-563, May 1976.
19. J.W. Modestino and A.H. Levesque, "A Mixture Model for VLF Noise with Applications to M-ary Spread-Spectrum Signaling," *Proc. of the Fifth Annual Princeton Conf. on Information Sciences and Systems*, (abstract only), March 25-26, 1971.

20. J.W. Tukey, "A Survey of Sampling from Contaminated Distributions," in *Contributions to Probability and Statistics*, I. Olkin, ed., Stanford Univ. Press, pp. 448-485, 1960.

21. D.F. Andrews, P.J. Bickel, F.R. Hampel, P.J. Huber, W.H. Rogers, J.W. Tukey, *Robust Estimates of Location: Survey and Advances*, Princeton University Press, Princeton, NJ, 1972.

22. E.J. Modugno, III, *The Detection of Signals in Impulsive Noise*, Ph.D. dissertation, Dept. of EECS, Princeton University, Princeton, NJ, 1982.

23. S.C. Schwartz, "A Series Technique for the Optimum Detection of Stochastic Signals in Noise," *IEEE Trans. Inform. Th.*, Vol. IT-15, pp. 362-369, May 1969.

24. J. Capon, "On the Asymtotic Efficiency of Locally Optimum Detectors," *IRE Trans. Inform. Theory*, Vol. IT-7. pp. 67-71, April 1961.

25. E.J.G. Pitman, *Some Basic Theory for Statistical Inference* London:Chapman and Hall, 1979.

26. C.W. Helstrom, *Statistical Theory of Signal Detection*, Oxford:Pergamon Press, 1968.

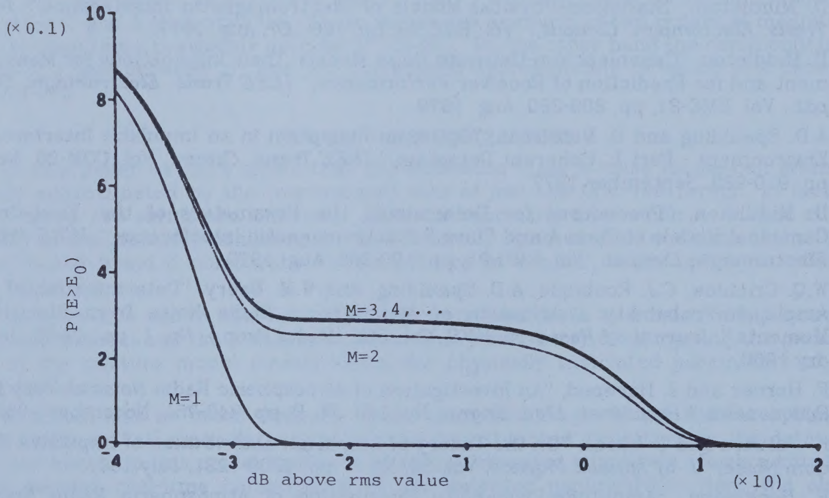


Figure 1. Truncated envelope distribution for M=1,2,3,...
(A=.35,Γ=.0005).

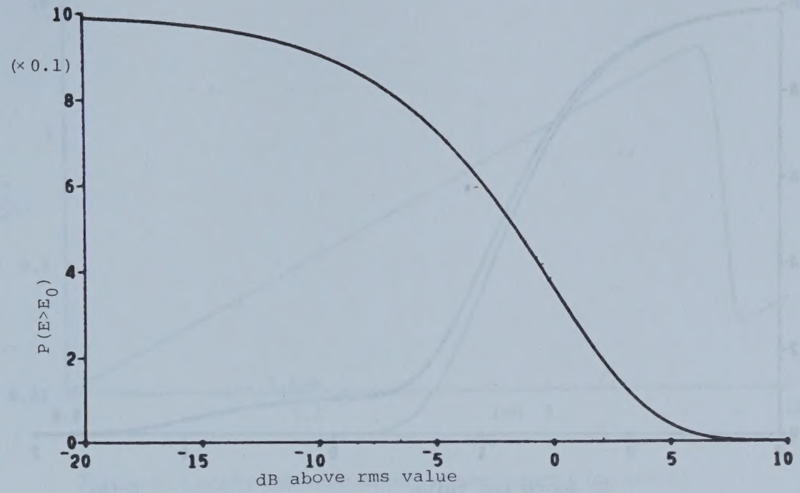


Figure 2. Truncated envelope distribution for M=1,2,3,...
(A=.0001,Γ=50).

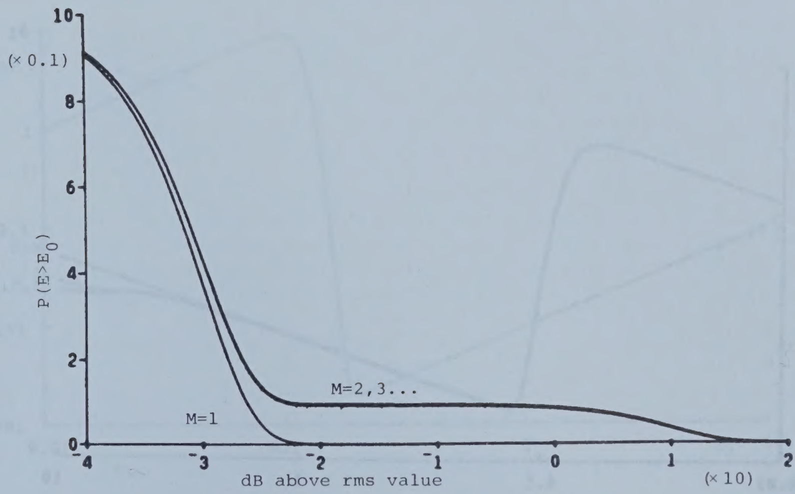


Figure 3. Truncated envelope distribution for M=1,2,3,...
(A=.1,Γ=.001).

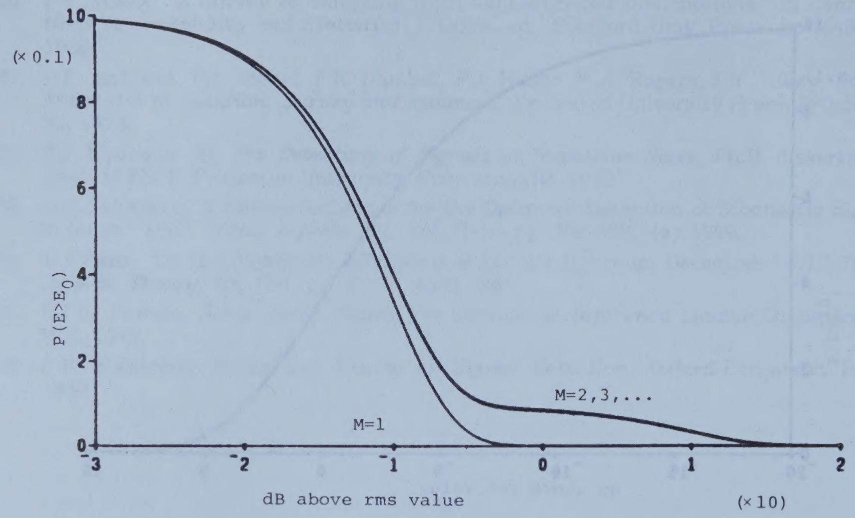


Figure 4. Truncated envelope distribution for M=1,2,3,... (A=.1,Γ=.1).

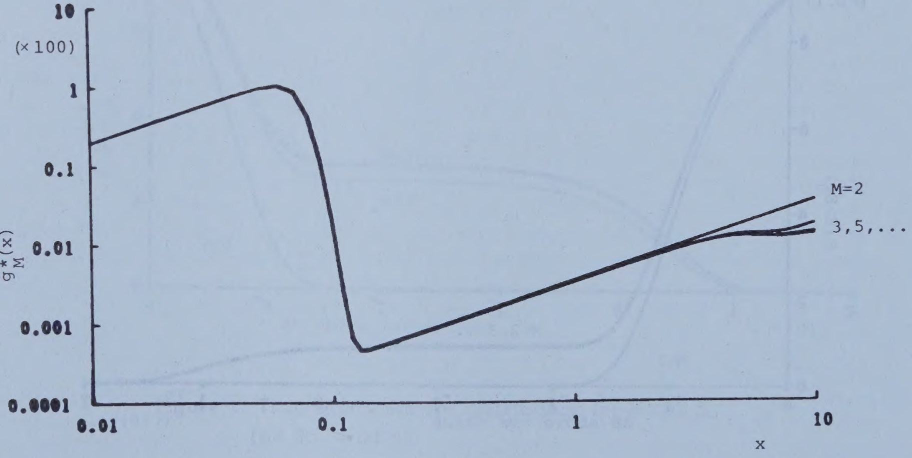


Figure 5. Locally optimum detector nonlinearity for noise PDF f_M, M=2,3,4,... (A=.35,Γ=.0005).

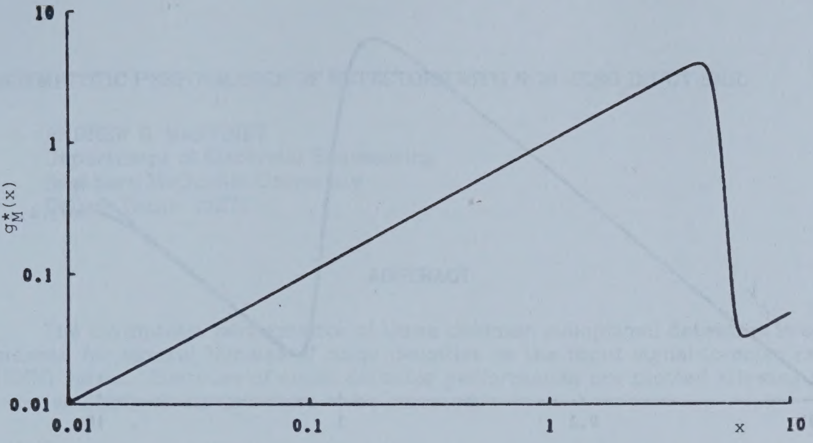


Figure 6. Locally optimum detector nonlinearity for noise PDF f_M, M=2,3,4,... (A=.0001,Γ=50).

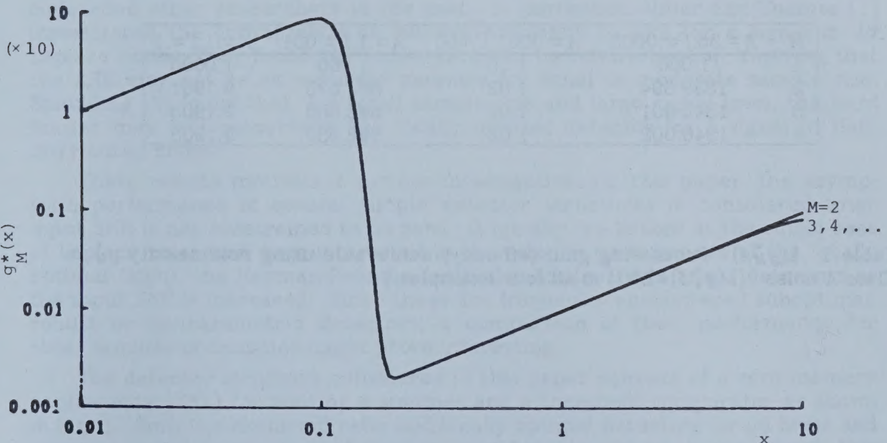


Figure 7. Locally optimum detector nonlinearity for noise PDF f_M, M=2,3,4,... (A=.1,Γ=.001).

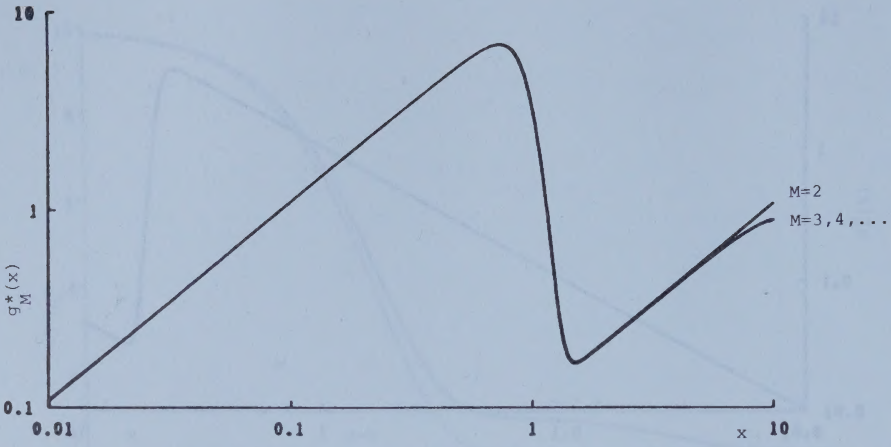


Figure 8. Locally optimum detector nonlinearity for noise PDF f_M, M=2,3,4,... (A=.1,Γ=.1).

M	A=.35,Γ=.0005	A=.0001,Γ'=50	A=.1,Γ=.001	A=.1,Γ=.1
1	1.000	1.00	1.000	1.0000
2	1339.894	1.02	892.578	9.1901
3	1340.001	1.02	892.580	9.1903
4	1340.002	1.02	892.580	9.1903

Table 1. $L(g^*_M, f)$. Processing gain (efficacy) achievable using nonlinearity g^*_M in Class A noise. ($L(g^*_4, f)=L^*(f)$ in all four examples.)

ASYMPTOTIC PERFORMANCE OF DETECTORS WITH NON-ZERO INPUT SNR.

ANDREW B. MARTINEZ
Department of Electrical Engineering
Southern Methodist University
Dallas, Texas 75275

ABSTRACT

The asymptotic performance of three common suboptimal detectors is considered for several families of noise densities as the input signal-to-noise ratio (SNR) varies. Contours of equal detector performance are plotted allowing the relative utility of the detectors to be assessed.

I. INTRODUCTION

Frequently, the performance of locally optimal and suboptimal detection schemes is evaluated using asymptotic measures such as the Asymptotic Relative Efficiency (ARE). Unfortunately, the ARE requires the assumption, not only of a large number of samples (for the central limit theorem to hold), but of a vanishingly small input signal-to-noise ratio (SNR). It follows that any optimal or suboptimal system designed by maximizing ARE may be optimal only for a signal level of zero, and might in fact perform quite poorly for non-zero SNR.

From a survey of the literature, it is evident that these assumptions have concerned other researchers in the past. In particular, Miller and Thomas [1] investigated the convergence of Relative Efficiency to ARE for a signal in iid Laplace noise. They found the convergence to be relatively slow, implying that the ARE may not be an accurate measure for small to moderate sample size. Spaulding [2] found that, for small sample size and large signal level, the hard limiter may well outperform the locally optimal detector for a signal in Hall-distributed noise.

These results motivate a further investigation. In this paper, the asymptotic performance of several simple detector structures is considered when input SNR is not constrained to be zero. Originally, we looked at the single case of Laplace noise and were interested in the relative performance of the locally optimal (sign), the Neyman-Pearson (amplifier limiter), and linear detectors as the input SNR is increased. Since these are frequently encountered suboptimal, robust or nonparametric detectors, a comparison of their performance for other families of densities might prove interesting.

The detector structure considered in this paper consists of a zero memory nonlinearity (ZNL) followed by a summer and a threshold comparator as shown in Fig. 1. Both the likelihood ratio and locally optimal detectors for iid noise and constant signal have this configuration, and for this reason among others, this form has particular appeal not only in optimal but in suboptimal detector design. In this paper, three detector ZNL's are used: the linear amplifier, the sign detector and the amplifier limiter.

II. DETECTION PROBLEM

The problem of detecting a constant signal in additive noise, given a sequence of observations, can be expressed as an hypothesis and an alternative:

$$\mathrm{H}_0\text{: } x_i = n_i$$
$$\mathrm{H}_1\text{: } x_i = n_i + \vartheta$$

Under the hypothesis H_0, the observation x_i consists of noise only; under the alternative H_1 it consists of noise plus an additive signal ϑ. When the noise is assumed to be independent and identically distributed (iid) with density $f(n_i)$, both the likelihood ratio detector and the locally optimal detector consist of a ZNL $g(\cdot)$ followed by a linear filter. If the number of samples is m then the test statistic T_m is given by

$$T_m = \sum_{i=1}^{m} g(x_i)$$

To simplify notation somewhat, let

$$\mu_0 = E_0 T_m \qquad \sigma_0^2 = E_0 T_m^2 - E_0^2 T_m$$

and

$$\mu_\vartheta = E_\vartheta T_m \qquad \sigma_\vartheta^2 = E_\vartheta T_m^2 - E_\vartheta^2 T_m$$

where E_0 and E_ϑ represent the expected value under H_0 and H_1 respectively.

If m is sufficiently large for the central limit theorem to be applied, then under H_0 and H_1 the two random variables

$$\frac{T_m - \mu_0}{\sqrt{m}\,\sigma_0} \quad \text{and} \quad \frac{T_m - \mu_\vartheta}{\sqrt{m}\,\sigma_\vartheta}$$

are both approximately unit normal, and the approximate level α is therefore

$$\alpha = \int_{\frac{t - m\mu_0}{\sqrt{m}\,\sigma_0}}^{\infty} d\Phi(x) = \Phi\left(\frac{m\mu_0 - t}{\sqrt{m}\,\sigma_0}\right)$$

where Φ is the Gaussian N(0,1) cdf. The threshold t can be found in terms of α:

$$t = m\mu_0 - \sqrt{m}\,\sigma_0 \Phi^{-1}(\alpha)$$

Likewise the approximate power can be computed:

$$\beta = \int_{\frac{t - m\mu_\vartheta}{\sqrt{m}\,\sigma_\vartheta}}^{\infty} d\Phi(x) = \Phi\left(\frac{m\mu_\vartheta - t}{\sqrt{m}\,\sigma_\vartheta}\right)$$

Using the expression for the threshold yields the power function:

$$\beta(\alpha) = \Phi\left(\frac{\sigma_0}{\sigma_\vartheta}\Phi^{-1}(\alpha) + \sqrt{m}\,\frac{\mu_\vartheta - \mu_0}{\sigma_\vartheta}\right)$$

The approximate power of a test for fixed level α and large sample size m is a monotone increasing function of the square root of the detector output SNR, where

$$SNR_{OUT} = \frac{(\mu_\vartheta - \mu_0)^2}{\sigma_\vartheta^2}$$

As $\sqrt{m} \to \infty$, the term containing the square root of the SNR will clearly dominate the other term in the power function, and the SNR is an accurate measure of detector performance. For moderate m, where the Gaussian assumption may still be quite good, the term $\sigma_0 \Phi^{-1}(\alpha)/\sigma_\vartheta$ can become important. In this case, using the SNR to compare the performance of two tests may not be enough; $\sigma_0/\sigma_\vartheta$ should ideally equal the same constant for both tests. In the following examples $\sigma_0 = \sigma_\vartheta$ for all detectors considered. Asymptotically, the results are equivalent, but this should yield more valid results for moderate m.

III. COMPARISON OF SEVERAL TESTS

In this section, three tests are considered: the linear detector, sign detector and the amplifier limiter. Under H_0 the observation x has distribution $F(x)$ and under H_1 distribution $F(x-\vartheta)$. Obviously, the noise variance will have no effect on the gain in SNR of each detector (input to output), and therefore it can be set equal to unity with no loss of generality.

1. The linear detector (LD) is simple and frequently used. It is usually justified by making a Gaussian assumption about the noise, and is both the Neyman Pearson and locally optimal detector for Gaussian noise. The LD has the ZNL

$$g(x) = x$$

and so

$$\mu_0 = 0 \qquad \mu_\vartheta = \vartheta \quad \text{and} \quad \sigma_0^2 = \sigma_\vartheta^2 = 1$$

and the output SNR of this detector is

$$SNR_{LD} = \vartheta^2 \tag{1}$$

2. The sign detector (SD) is probably the simplest ZNL which may be effective in an impulse environment. In fact it has been shown [4] to perform reasonably well under certain circumstances.

The SD has the ZNL:

$$g(x) = sgn(x)$$

Thus

$$\mu_0 = 1 - 2F(0) \qquad \mu_\vartheta = 1 - 2F(-\vartheta) \quad \text{and} \quad \sigma_0 = \sigma_\vartheta = 1$$

and the SNR of the SD is

$$SNR_{SD} = 4[F(0) - F(-\vartheta)]^2 \tag{2}$$

3. The form of the amplifier limiter (AL) used here is the Neyman-Pearson optimal test ZNL for Laplace noise, with breakpoints at the signal level ϑ. While this is not necessarily the best breakpoint for all noise environments, it avoids the difficulties inherent in choosing optimal breakpoints [4] while still preserving some of the robust characteristics described by Martin and Schwartz [6].

The AL has the ZNL:

$$g(x) = \begin{cases} 0 & x < 0 \\ x & 0 \leq x \leq \vartheta \\ \vartheta & \vartheta \leq x \end{cases}$$

Thus

$$\mu_0 = \vartheta[1 - F(\vartheta)] + \int_0^\vartheta x \, dF(x)$$

$$\mu_\vartheta = \vartheta[1 - F(-\vartheta)] + \int_{-\vartheta}^{0} x \, dF(x)$$

$$\sigma_0{}^2 = \int_{0}^{\vartheta} x^2 \, dF(x) + \vartheta^2[1 - F(+\vartheta)] - \mu_0{}^2$$

$$\sigma_\vartheta = \int_{-\vartheta}^{0} x^2 \, dF(x) + 2\vartheta \int_{-\vartheta}^{0} x \, dF(x) + \vartheta^2[1 - F(-\vartheta)] - \mu_\vartheta{}^2$$

Denote the ith incomplete moment as I_i, where

$$I_i = \int_{0}^{\vartheta} x^i \, dF(x)$$

Then if f is assumed symmetric:

$$\mu_0 = \vartheta[\tfrac{1}{2} - I_0] + I_1$$

$$\mu_\vartheta = \vartheta[\tfrac{1}{2} + I_0] - I_1$$

$$\sigma_0{}^2 = \sigma_\vartheta{}^2 = I_2 + \vartheta^2[\tfrac{1}{2} + I_0] - 2\vartheta I_1 - \mu_\vartheta{}^2$$

The AL has SNR

$$SNR_{AL} = \frac{4(\vartheta I_0 - I_1)^2}{I_2 + \vartheta^2/4 - \vartheta I_1 - \vartheta^2 I_0{}^2 - I_1{}^2 + 2\vartheta I_0 I_1} \tag{3}$$

IV. RESULTS

A. GENERALIZED GAUSSIAN

The first distribution considered is the generalized Gaussian.

$$f(x) = \frac{\alpha^{1/c} c}{2\Gamma(1/c)} \exp(-\alpha |x|^c)$$

For a variance of unity

$$\alpha^{2/c} = \Gamma(3/c)/\Gamma(1/c)$$

In addition to including the Gaussian and Laplace as special cases, the generalized Gaussian offers considerable control over the rate of tail decay through decay parameter c. It has been suggested as a model of impulsive atmospheric noise, particularly for $0.1 < c < 0.6$, and used to demonstrate the locally optimal detector by Miller and Thomas [3], and later by Lu and Eisenstein [5].

The incomplete moments I_i (for $i = 0, 1, 2$) are

$$I_0 = \frac{\gamma(1/c, \alpha\vartheta^c)}{2\Gamma(1/c)}$$

$$I_1 = \frac{\gamma(2/c, \alpha\vartheta^c)}{2[\Gamma(1/c)\Gamma(3/c)]^{1/2}}$$

$$I_2 = \frac{\gamma(3/c, \alpha\vartheta^c)}{2\Gamma(3/c)}$$

where $\gamma(.,.)$ is the incomplete γ function.

Using the incomplete moments and Eqs. (1-3), the output SNR's of the LD, SD and AL detectors are computed and plotted in Figs. 2-6 for several values of decay parameter c. Two values of c are of particular interest.

When $c = 1$ (Fig. 3) the generalized Gaussian reduces to the Laplace density. The SD is then locally optimal and the AL is Neyman-Pearson optimal. As expected, the AL outperforms both the LD and SD for when the input SNR is greater than zero. Perhaps surprisingly, the performance of the locally optimal SD decays significantly as input SNR increases. In fact, the LD outperforms the SD when $SNR_{IN} > 0.272$.

The Gaussian density results when $c = 2$ (Fig. 5.). In this case, the LD is both locally and Neyman-Pearson optimal, and as expected, it outperforms both the AL and the SD.

The points of greatest interest in Figs. 2-6 are the crossover points at which two tests have equal output SNR. In Fig. 7, the contours of $SNR_{SD} = SNR_{LD}$ and $SNR_{AL} = SNR_{LD}$ are plotted in the $c \times SNR_{IN}$ plane. The two contours divide the portion of the plane considered into three regions.

When $SNR_{IN} = 0$, the SD and AL are equivalent; for all $SNR_{IN} > 0$, the AL outperforms the SD. As configured here, the AL requires the signal level ϑ to be known, and when ϑ is known the AL would normally be used over the SD. The LD is to be preferred over the AL only in the rightmost region, right of the contour $SNR_{AL} = SNR_{LD}$. When ϑ is not known, the choice is between the LD and the SD. The LD outperforms the SD in all but the region left of the contour $SNR_{SD} = SNR_{LD}$.

In Fig. 8, the the relative performance of the SD and the LD is shown as contours of $(SNR_{SD}/ SNR_{LD})_{dB}$ in the $c \times SNR_{IN}$ plane. The $0dB$ line corresponds to the line of equal performance in Fig. 7, and performance of the SD relative to that of the LD can be seen to fall off quickly not only with the transition from Laplace ($c = 1$) to Gaussian ($c = 2$) but with increases in SNR_{IN}.

In practice, one might use single or (especially in nonstationary noise environments) multiple estimates of c and SNR_{IN} and Fig. 7 to choose a detector structure. To simplify the process, since c is rather difficult to estimate, the two contours $SNR_{SD} = SNR_{LD}$ and $SNR_{AL} = SNR_{LD}$ are given in the $\beta_2 \times SNR_{IN}$ plane in Fig. 9. Here β_2 is the normalized fourth moment, or kurtosis of the noise, frequently used as a measure of density "peakedness". The kurtosis is defined to be:

$$\beta_2 = \mu_4 / \mu_2^2$$

where μ_2 and μ_4 are the second and fourth noise moments about the mean. Not only is the kurtosis β_2 easier to estimate, but it is a more general measure than c and can be applied to many other densities.

B. PEARSON VII

The next model considered is the Pearson VII density:

$$F(x) = K(1 + \alpha x^2)^{-m}$$

where

$$K = \frac{\alpha^{1/2}\Gamma(m)}{\Gamma(1/2)\Gamma(m - 1/2)} \qquad \alpha = \frac{\beta_2 - 3}{2\mu_2\beta_2} \qquad m = \frac{5\beta_2 - 9}{2(\beta_2 - 3)}$$

Rather than exponentially decaying tails as in the previous example, the Pearson VII density has tails which decay algebraically. In the limit as $\beta_2 \rightarrow 3$ the density approaches the Gaussian density.

In this case the incomplete moments I_i (for $i = 0, 1, 2$) are computed numerically. Using these incomplete moments and Eqs. (1-3), the output SNR's of the LD, SD and AL were found for several values of kurtosis, β_2, and plotted in Figs. (10-13).

At $\beta_2 = 3$, the Gaussian density results, and the performance of the three detectors is described in the last section, and shown in Fig. 5. As β_2 increases, the density becomes heavier tailed and the performance of both the AL and the SD improves relative to that of the LD. As before, the AL consistently outperforms the SD.

In Fig. 14, the contour of equal performance of the SD and LD, $SNR_{SD} = SNR_{LD}$, is shown in the $\beta_2 \times SNR_{IN}$ plane. Considering the results of the last section, it is interesting how poorly the SD performs in this particular noise environment. Comparing Figs. 9 and 14, not only is the SD for Pearson VII noise much more sensitive to increases in SNR_{IN} but the kurtosis must be nearly three times as great (at $SNR_{IN} = 0$) for the SD to outperform the LD.

C. MIXTURE MODEL

The next noise model considered is a mixture model. Under certain conditions [4,6,7] a noise process may be well-described by a mixture of the form

$$f(x) = (1-\varepsilon) f_B(x) + \varepsilon\, f_I(x)$$

where f_B describes a background noise process contaminated by bursts of impulsive f_I distributed noise. In the examples which follow, the background noise is assumed Gaussian:

$$f_B(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left\{-\frac{x^2}{2\sigma^2}\right\}$$

The impulsive contaminant is assumed to be Laplace or Gaussian.

1. Gauss-Laplace mixture

The Gauss-Laplace mixture assumes the contaminating impulsive noise to be Laplace:

$$f_I(x) = \frac{\alpha}{2} \exp(-\alpha |x|)$$

The ratio of the variances $\gamma = 2/\alpha^2\sigma^2$ proves to be an important constant.

The incomplete moments, I_i can be computed numerically, and the resulting SNR's can be found from Eqs (1-3). The contours of equal SNR for the linear and sign detector are plotted for several values of γ in Fig. 15. The SD outperforms the LD below the contours, and the LD is superior above. Typically γ will be large and ε small corresponding to infrequent large energy bursts. In this case, the sign detector may be of use as shown by the $\gamma = 100$ contour.

2. Gauss-Gauss Mixture

In this case the contaminant is assumed also to be Gaussian, with variance s^2. The ratio of the variances again proves useful, where $\gamma = s^2/\sigma^2$.

The contours of $SNR_{SD} = SNR_{LD}$ are plotted in Fig. 16 for several values of γ. The SD outperforms the LD below the contours as in the last example. It is interesting to note that while the LD is optimal at $\varepsilon = 0$ or $\varepsilon = 1$ since f is then purely Gaussian, the performance of the SD may in fact surpass that of the LD for $0 < \varepsilon < 1$. As stated above, γ will frequently be large and ε small; in this case the SD would appear to perform quite well.

V. CONCLUSIONS

Care should be taken in applying locally optimal results, since the locally optimal detector is designed for zero input SNR. As shown in the Laplace example, the performance can deteriorate rather quickly as SNR_{IN} increases, so quickly in fact that the linear detector may well outperform the locally optimal detector for SNR_{IN} fairly small.

The comparison of generalized Gaussian and Pearson VII results indicates the importance of correctly choosing a noise model. The sign detector performs considerably less well, particularly in sensitivity to increases in SNR_{IN}, for Pearson VII distributed noise. From these results, we can conclude that even two densities which have equal moments up to fourth order can yield decidedly different performance from the same detectors.

In general the performance of the amplifier limiter is quite good for the noise models used. The sign detector on the other hand is at times disappointing. One case in which it does appear to work well was for the mixture model with infrequent but high energy impulsive bursts (γ large, ε small). Not only does the sign detector outperform the linear detector, but it is relatively insensitive to the form of the contaminant; similar performance is seen for both Laplace and Gaussian contaminant.

Acknowledgement

This research is supported by the National Science Foundation under Grant ECS-8206090 and by the Office of Naval Research under Contract N00014-81-K-0146.

REFERENCES

1. J.H. Miller and J.B. Thomas, "Numerical Results on the Convergence of Relative Efficiencies," *IEEE Trans. on Aerospace and Electronic Systems*, vol. AES-11, no. 2, March 1975.

2. A.D. Spaulding, "Locally Optimum and Suboptimum Detector Performance in Non-Gaussian Noise," *Proc. of IEEE ICC 1982*, vol. I, pp. 2H.2.1-7, June 1982.

3. J.H. Miller and J.B. Thomas, "Detectors for Discrete-Time Signals in Non-Gaussian Noise," *IEEE Trans. on Inform. Theory*, vol. IT-18, no. 2, pp. 241-250, March 1977.

4. J.H. Miller and J.B. Thomas, "Robust Detectors for signals in Non-Gaussian Noise," *IEEE Trans. on Comm.* vol. COMM-25, no. 7, pp. 686-690, July 1977.

5. N. H. Lu and B. A. Eisenstein, "Detection of Weak Signals in Non-Gaussian Noise," *IEEE Trans. on Inform. Theory*, vol. IT-27, no. 6, pp. 755-771, November 1981.

6. R.D. Martin and S.C. Schwartz, "Robust detection of a known Signal in Nearly Gaussian Noise," *IEEE Trans. on Inform. Theory*, vol. IT-17, pp.50-56, January 1971.

7. W.J. Richter, Jr. and T.I. Smits, "Signal Design and Error Rate for an Impulsive Noise Channel," *IEEE Trans. on Comm. Tech.*, vol. COMM-19, pp.446-458, August 1971.

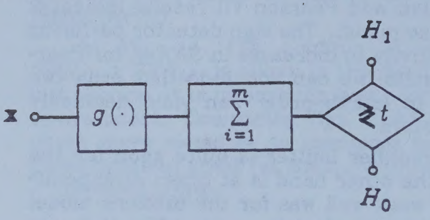


Fig. 1. Typical detector structure.

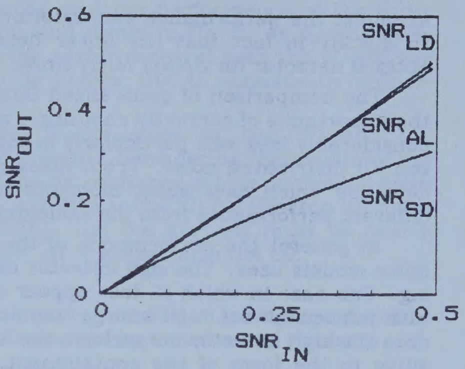


Fig. 4. SNR_{IN} versus SNR_{OUT} for generalized Gaussian, c = 1.5.

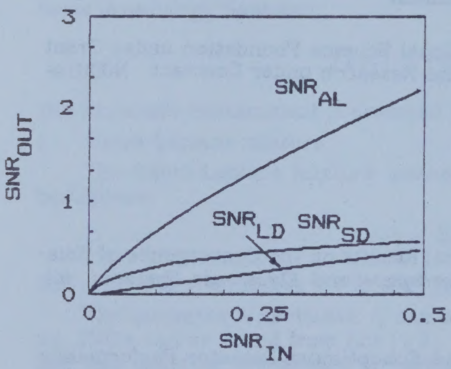


Fig. 2. SNR_{IN} versus SNR_{OUT} for generalized Gaussian, c = 0.5.

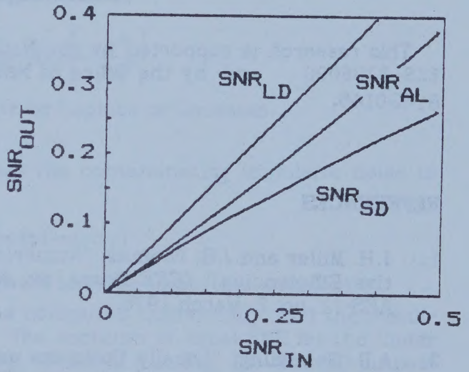


Fig. 5. SNR_{IN} versus SNR_{OUT} for generalized Gaussian, c = 2.0.

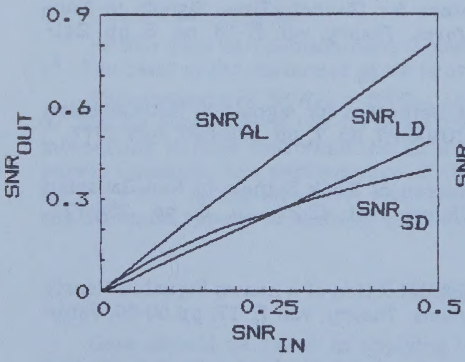


Fig. 3. SNR_{IN} versus SNR_{OUT} for generalized Gaussian, c = 1.0.

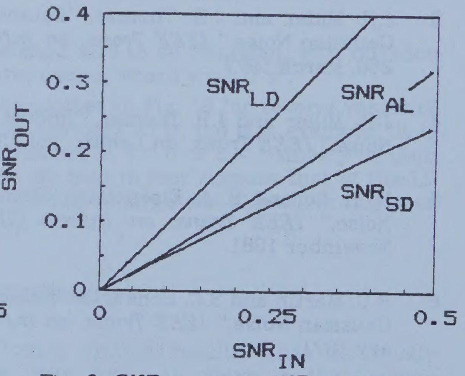


Fig. 6. SNR_{IN} versus SNR_{OUT} for generalized Gaussian, c = 2.5.

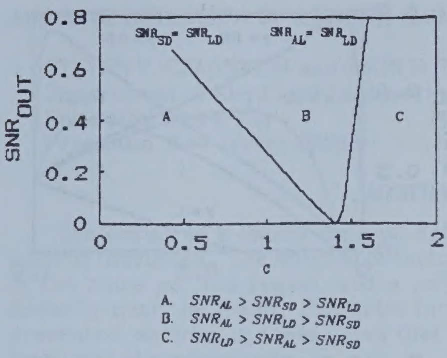


Fig. 7. Contours of $SNR_{SD} = SNR_{LD}$ and $SNR_{AL} = SNR_{LD}$ in the $c \times SNR_{IN}$ plane for generalized Gaussian noise.

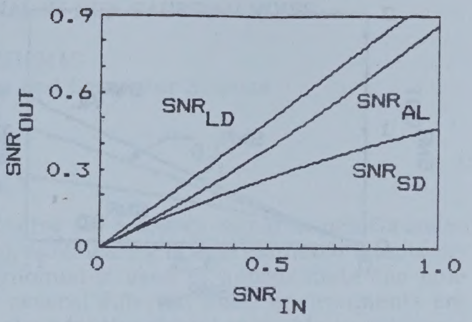


Fig. 10. SNR_{IN} versus SNR_{OUT} for Pearson VII, $\beta_2 = 3.1$.

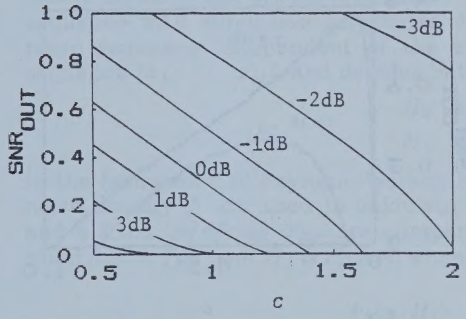


Fig. 8. Contours of $(SNR_{SD} / SNR_{LD})_{dB}$ in the $c \times SNR_{IN}$ plane for generalized Gaussian noise.

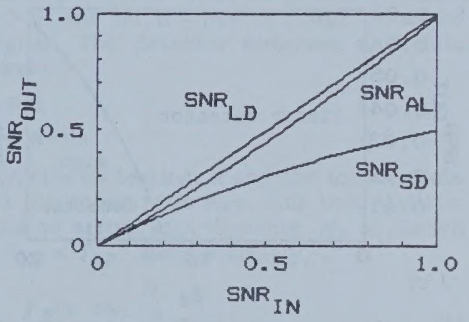


Fig. 11. SNR_{IN} versus SNR_{OUT} for Pearson VII, $\beta_2 = 4.0$.

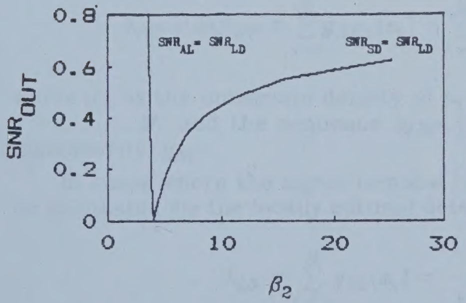


Fig. 9. Contours of $SNR_{SD} = SNR_{LD}$ and $SNR_{AL} = SNR_{LD}$ in the $\beta_2 \times SNR_{IN}$ plane for generalized Gaussian noise.

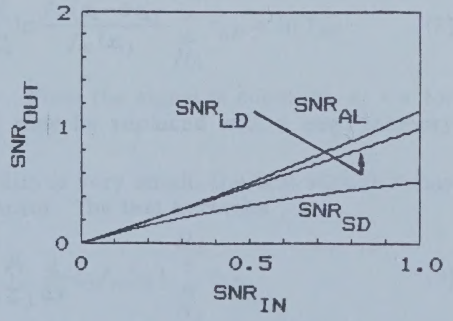


Fig. 12. SNR_{IN} versus SNR_{OUT} for Pearson VII, $\beta_2 = 6.0$.

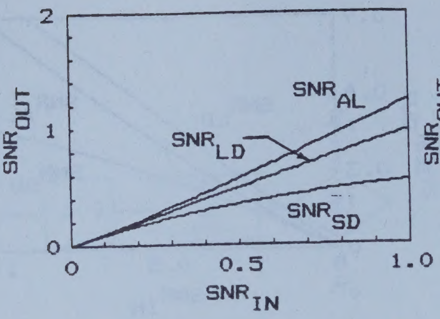


Fig. 13. SNR_{IN} versus SNR_{OUT} for Pearson VII, $\beta_2 = 10$.

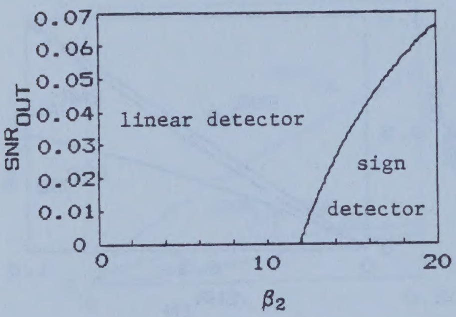


Fig. 14. Contour of SNR_{SD} = SNR_{LD} in $\beta_2 \times SNR_{IN}$ for Pearson VII.

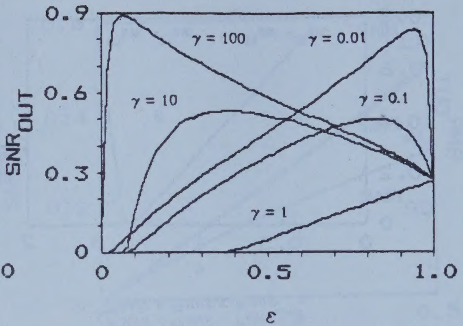


Fig. 15. Contours of SNR_{SD} = SNR_{LD} in $\varepsilon \times SNR_{IN}$ for Gauss-Laplace mixture.

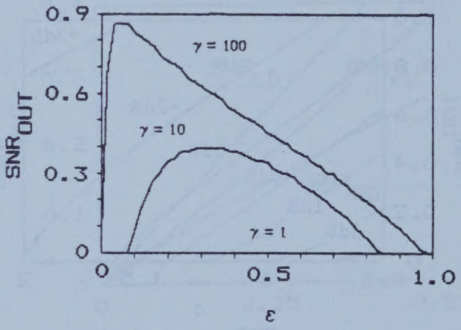


Fig. 16. Contours of SNR_{SD} = SNR_{LD} in $\varepsilon \times SNR_{IN}$ for Gauss-Gauss mixture.

ADAPTIVE DETECTION OF A KNOWN SIGNAL IN NON-GAUSSIAN NOISE

STEVEN V. CZARNECKI and JOHN B. THOMAS
Department of Electrical Engineering and Computer Science
Princeton University
Princeton, New Jersey 08540

ABSTRACT

The design of a locally optimal detector for a known signal in non-Gaussian noise is discussed. The optimal detector nonlinearity is approximated adaptively in the noise pdf tail region, and a polynomial is used to approximate the nonlinearity near the mean. Examples for several different noise environments are presented, showing in these cases that the adaptive detector is able to achieve a high level of performance

1. INTRODUCTION

A binary hypothesis test may be used to model the problem of detecting a known signal in the presence of noise. Let $\vartheta \mathbf{s} = \{s_1, \ldots, s_M\}$ be the known signal sequence with amplitude parameter $\vartheta > 0$, and let $\mathbf{n} = \{n_1, \ldots, n_M\}$ be an iid noise sequence independent of the signal. The detector observes $\mathbf{x}$, a data sequence $\{x_1, \ldots, x_M\}$, and decides between:

$$H_0: \ \mathbf{x} = \mathbf{n}$$

$$H_1: \ \mathbf{x} = \mathbf{n} + \vartheta \mathbf{s}$$

In the framework of Neyman-Pearson hypothesis testing, $\mathbf{x}$ and the multivariate noise density f are used to calculate a likelihood ratio Λ_{NP}. This test statistic and a fixed threshold T_{NP} are compared to arrive at a decision: H_1 is chosen when $\Lambda_{NP} > T_{NP}$, and H_0 is chosen when $\Lambda_{NP} \leq T_{NP}$. More precisely,

$$\Lambda_{NP} = \frac{f_{\mathbf{X}}(\mathbf{x} \mid H_1)}{f_{\mathbf{X}}(\mathbf{x} \mid H_0)} = \frac{f_{\mathbf{N}}(\mathbf{x} - \vartheta \mathbf{s})}{f_{\mathbf{N}}(\mathbf{x})} \underset{H_0}{\overset{H_1}{\gtrless}} T_{NP} \tag{1}$$

Since the noise is iid, and the logarithm function is monotonic, an equivalent test is

$$\lambda_{NP} = \ln \Lambda_{NP} = \sum_{i=1}^{M} g_{NP;i}(x_i) = \sum_{i=1}^{M} \ln \frac{f_n(x_i - \vartheta s_i)}{f_n(x_i)} \underset{H_0}{\overset{H_1}{\gtrless}} t_{NP} = \ln T_{NP} \tag{2}$$

where f_n is the univariate density of n_i. When the signal is constant, $s_i = s$ for $i = 1, \ldots, M$, and the sequence $\{g_{NP;i}\}$ may be replaced with a zero memory nonlinearity, g_{NP}.

In cases where the signal-to-noise ratio is very small, the test statistic may be calculated via the locally optimal detector. The test becomes

$$\lambda_{LO} = \sum_{i=1}^{M} g_{LO}(x_i) = -\sum_{i=1}^{M} \frac{\mathrm{d}}{\mathrm{dx}} \ln f_n(x_i) \underset{H_0}{\overset{H_1}{\gtrless}} t_{LO} \tag{3}$$

Implicit in both detection methods is a requirement that the noise pdf must be known exactly. In general, the noise statistics are not known with precision and the design of the LO or NP detector is not straightforward. Alternative detection strategies are available, and among these are (1) detectors which are robust with respect to deviations from a nominal noise environment [2,8]; (2) nonparametric detectors which use only very general information about the underlying noise distribution [3]; and (3) fixed or adaptive suboptimal detectors which have nearly optimal performance [4,7,13]. An additional consideration is that often the noise environment is nonstationary and an adaptive structure is necessary.

2. DETECTION IN NON-GAUSSIAN NOISE

The classical assumption in the design and analysis of detection systems is that the noise is Gaussian. This assumption has attractive features in that the Gaussian model is mathematically tractable, the optimal detector structure is linear, and strong justification for the model is available in the form of the Central Limit Theorem. Measurements of typical noise environments have led to the conclusion that the true noise distribution often is described better by a heavier tailed pdf [5,6,12,13]. This type of noise can be ascribed to a nominal Gaussian environment with a heavy tailed impulsive noise contaminant, and to the fact that a real noise comprises a finite sum of random events; convergence to the Gaussian pdf is not complete for a finite sum, but the sum pdf is most nearly Gaussian near the mean, with the tails converging to the Gaussian pdf only in the limiting case. For example, previous work in detection has taken note of non-Gaussian noise environments, where the noise density is approximately Gaussian near the mean, but heavier tailed away from the mean [1,6,10].

The NP and LO detector nonlinearities related to near-Gaussian heavy tailed densities are typically composed of a linear region surrounded by tails which compress, limit, or even blank large data observations. Work in robust estimation of the mean has similarly suggested that, in heavy tailed noise, a robust estimator should reduce the influence of very large data observations while leaving observations near the mean relatively unchanged [9].

3. A SUBOPTIMAL DETECTOR APPROXIMATION

This paper presents an approach to the design of a noise-adaptive suboptimal detector with these ideas in mind. Attention is focused on the case of LO detection of a constant binary signal in discrete time, with iid noise assumed. Further, the noise pdf is restricted to be unimodal, symmetric about its mean placed at the origin, and to have nonzero support over the entire real line. We are interested in approximating ZNL's loosely specified by the following characteristics:

a) continuous, with continuous low-order derivatives
b) approximately linear at the origin
c) odd symmetric about the origin
d) strictly positive to the right of the origin
e) monotone in the tail regions
f) possess only one local extremum on either side of the origin

Pdf's which are non-Gaussian in the sense previously described can generate LO nonlinearities with these characteristics, which are typical of robust and suboptimal detector nonlinearities, and also of many of the influence curves given for robust estimation procedures [11].

The tail behavior of these LO nonlinearities range from linear for a noise pdf with Gaussian tails, to a limiter for exponentially decreasing pdf tails, to a blanker for algebraically decreasing pdf tails. In general, the heavier tailed the noise density is relative to the Gaussian pdf, the more severely curtailed is the effect of large data observations.

The objective of a noise adaptive nonlinearity, then, should be to relate the ZNL tail behavior to the actually observed noise pdf tail behavior. We propose the following method: It has been reported [12] that the generalized Gaussian density

$$f_c(x) = \frac{\eta c}{2\Gamma(1/c)} \exp(-|\eta x|^c) \tag{4}$$

in certain instances can describe the pdf tails of some physical noise sources. For a noise variance of σ^2, the parameter η is defined by

$$\eta = \left[\frac{\Gamma(3/c)}{\sigma^2\Gamma(1/c)}\right]^{1/2}$$

The corresponding LO nonlinearities, shown in Figure 1, can be written as

$$g_{LO}(x) = c\eta^c\,|x|^{c-1}\operatorname{sgn}(x) \tag{5}$$

with the c conveniently parameterizing ZNL tail behavior. Therefore, we model the observed noise pdf tails via the generalized Gaussian family. The suboptimal LO nonlinearity thus will have power law tails described by

$$\hat{g}_{LO}(x) = \hat{c}\eta^{\hat{c}}\,|x|^{\hat{c}-1}\operatorname{sgn}(x) \quad \text{for}\,|x| > x_0 \tag{6}$$

It is necessary to find a value $\hat{c}$ such that $f_{\hat{c}}$ is a good approximant to the tail behavior of f_n, the true but unknown underlying noise density. A simple way to do this is to equate the tail probability of $f_{\hat{c}}$ with the observed tail mass

$$\hat{P}_T = \frac{1}{N}\sum_{i=1}^{N} I_{(T,\infty)}(|r_i|) \tag{7}$$

Here, I is the indicator function and r_i are the noise observations presumed available from a noise reference channel. The exponent c can be estimated then by finding $\hat{c}$ such that

$$2\int_T^{\infty} f_{\hat{c}}(x)dx = \hat{P}_T \tag{8}$$

For the purpose of calculating η, the observed noise variance $\sigma_N{}^2$ is used. The estimate $\hat{c}$ is defined by the implicit integral in (8), so it is desirable to derive a simpler explicit relationship

$$\hat{c} = h_T(\hat{P}_T) \tag{9}$$

One obvious method is to first calculate P_T as a function of c, and then interpolate this tabulation to approximate (9). This tabulated version of h_T is shown in Figure 2.

With σ^2 fixed and c small, the value of η, a scale factor, becomes large. Therefore, even though $f_c(x)$ approaches zero asymptotically at a much slower rate than the Gaussian pdf as $|x|$ gets large, the total probability mass in the tails is quite small. As a result, h_T is multiple valued, the density is extremely peaked, and the LO nonlinearity has a discontinuity at the origin. Clearly, for $c < 1$, the requirement of near-linearity at the origin is not met by (5). Since the objective in using the generalized Gaussian pdf is to relate the tail heaviness of an observed noise to a parameter governing the shape of the ZNL tail, we replace the anomalous behavior of the true function h with a simple linear relation

$$\hat{h}_T(\hat{P}_T) = a\hat{P}_T + b \tag{10}$$

where a and b are chosen to approximate (9) for a particular value of T. This approximation is plotted as the broken line on Figure 2. The values for a and b are chosen so that when $\hat{P}_T$ corresponds to Gaussian or exponential noise tails, (10) gives $\hat{c} = 2$ and $\hat{c} = 1$, respectively. Note that the linear relation allows the value of $\hat{c}$ to be negative for large tail probabilities.

The tail measurement threshold T must be chosen prior to estimating parameters a and b. One way to pick T is to choose a value which minimizes

$$\mathrm{E}_c\left\{\mathrm{var}\,\hat{c}\right\} = a^2\mathrm{E}_c\left\{\frac{P_T(1-P_T)}{N}\right\} \tag{11}$$

where N is the number of noise observations. For c uniformly distributed on the interval $[1,2)$, and P_T the tail probability of $f_c(x)$ for $|x| > T$, the constant $T = 3\sigma$ approximately minimizes (11).

The LO nonlinearities of the generalized Gaussian family have desirable tail behavior, but for small values of $\hat{c}$ the behavior does not meet the constraint of linearity near the origin. To eliminate this behavior, the ZNL needs modification in the region near the origin. A way in which to do this is to replace $\hat{g}_{LO}(x)$, for x near zero, with a function that fits the given desired characteristics (a-f). A suitable family of functions are polynomials $p(x)$ with the following characteristics:

$$p(x) = \alpha_3x^3+\alpha_2x^2+\alpha_1x+\alpha_0 \qquad \text{for} \quad 0 \le x \le x_0$$
$$p(0) = 0$$
$$p(|\pm x_0|)\mathrm{sgn}(\pm x_0) = \hat{g}_{LO}(\pm x_0)$$
$$p'(|\pm x_0|) = \hat{g}'_{LO}(\pm x_0)$$
$$p''(|\pm x_0|)\mathrm{sgn}(\pm x_0) = \hat{g}''_{LO}(\pm x_0)$$

The choice of tail behavior via $\hat{c}$ and the point x_0 completely specify $p(x)$. The method for choosing $\hat{c}$ has already been specified, so the choice of x_0 is the remaining free parameter.

A method equivalent to choosing the proper x_0 is to choose an arbitrary x_0 and scale the input to the ZNL with a factor ν. It is reasonable to choose ν to maximize the efficacy of the ZNL. For an arbitrary nonlinearity q, efficacy as a function of ν can be defined as

$$\xi_q(\nu) = \frac{\nu^2\mathrm{E}_f^2[q'(\nu x)]}{\mathrm{E}_f[q^2(\nu x)]-\mathrm{E}_f^2[q(\nu x)]} \tag{12}$$

where E_f refers to the expectation with respect to the density function $f(x)$, and q' refers to the first derivative of q.

At this point, specification of the suboptimal nonlinearity $\hat{g}$ is complete, and can be written as

$$\hat{g}(\nu x) = \begin{cases} p(|\nu x|)\,\mathrm{sgn}(\nu x) & \text{if} \quad |\nu x| \le x_0 \\ \hat{c}\,|\nu x|^{\hat{c}-1}\,\mathrm{sgn}(\nu x) & \text{if} \quad |\nu x| > x_0 \end{cases}; \tag{13}$$

Figures 3a-3c gives some examples of the types of nonlinearities available using this approximation method.

4. EXAMPLES

Analytic

We will now present examples of the use of $\hat{g}$ in approximating some known optimal LO nonlinearities.

The first comparison is between the approximate and exact versions of LO nonlinearities for the generalized Gaussian family. The exponent $\hat{c}$ is given by (10). after using the exact value c in (8) to obtain P_T. Since this is an analytical example, and the true noise density is known, numerical methods can be used to obtain ν^*, the value of ν which maximizes $\xi_g(\nu)$. The performance of the suboptimal ZNL relative to the LO nonlinearity can be measured by asymptotic relative efficiency, which is given easily in terms of efficacy as

$$\mathrm{ARE}_{\hat{g},g_{LO}} = \frac{\xi_{\hat{g}}}{\xi_{g_{LO}}} \tag{14}$$

Figure 4 compares the performance of $\hat{g}$, the LO detector and a linear detector (ld), in terms of $\mathrm{ARE}_{\hat{g},g_{LO}}$ and $\mathrm{ARE}_{g_{LO},ld}$. The suboptimal nonlinearity performs quite well for the range $1 \leq c \leq 2$, but for $c < 1$, performance deteriorates. This is easily explainable, since for small c, the LO nonlinearity output approaches $\pm\infty$ for inputs near zero, while the approximation method requires $\hat{g}$ to pass through the origin

Another family of heavy tailed pdf's is the Johnson S_u family. If X is distributed as an $N(0,\sigma^2)$ random variable, and we define a new random variable

$$y = \lambda \sinh\left[\frac{x}{\delta}\right] \tag{15}$$

then the density of Y has unity variance, and belongs to the Johnson S_u family, given by

$$f_\delta(y) = \frac{\delta}{\lambda\sqrt{2\pi}}\left[\left[\frac{y}{\lambda}\right]^2+1\right]^{-\frac{1}{2}} \exp\left[-\frac{\delta}{2}\sinh^{-1}\left[\frac{y}{\lambda}\right]^2\right] \tag{16}$$

with

$$\lambda = \left[\frac{2\sigma^2}{\exp(2/\delta^2)-1}\right]^{\frac{1}{2}} \tag{17}$$

The parameter δ controls the tail heaviness. As $\delta \rightarrow \infty$, the pdf tails become progressively lighter, and approach Gaussian tails in the limit.

Since f_δ is given and known, $\hat{P}_T$ can be calculated from (8), and (10) gives $\hat{c}$. Again, numerical methods can find the ν^* which maximizes (12). Some representative LO nonlinearities and suboptimal approximations are given for various values of δ in Figure 5, and Figure 6 presents the performance comparison of $\hat{g}$, g_{LO}, and ld. For this set of densities, the approximation method works quite well. Over the range $.8 \leq \delta \leq \infty$, the minimum of $\mathrm{ARE}_{\hat{g},g_{LO}}$ is .989, (occurring for $\delta = .8$). This means that only a small performance penalty would be incurred if $\hat{g}$ were to replace the LO detector. As a final comment, it should be observed that unlike the generalized Gaussian family, the Johnson S_u family fulfills the characteristics of a near Gaussian pdf given earlier.

A third family of heavy tailed densities was also investigated. This third family is the ε-contaminated Gaussian mixture, and in this case, the Gaussian-Gaussian mixture. The pdf of this noise can be written as

$$f_\varepsilon(x) = (1-\varepsilon)f_0(x) + \varepsilon f_1(x) \tag{18}$$

where f_0 represents the pdf of an $N(0,1)$ distributed random variable, and f_1 represents the pdf of an $N(0,\sigma^2)$ random variable, with σ^2 large. The parameter ε controls the degree to which f_1 contaminates the nominal density f_0, and is typically taken to be small. The performance of the suboptimal ZNL was calculated for some specific cases, and the results appear in Table 1. Figure 7 shows the comparison in the form of two LO nonlinearities and the corresponding approximations. The performance of the approximations in this case is not as good as the approximations under the Johnson S_u family. Once again not all the assumptions are met by the ε-contaminated Gaussian-Gaussian mixture on the form of the LO nonlinearity, as there is more than one local extremum on each side of the origin.

Table 1

Performance of g_{LO} and $\hat{g}$ in ε-contaminated Gaussian-Gaussian noise

σ^2	ε	x_0	$\hat{c}$	ν^*	$\mathrm{ARE}_{\hat{g},g_{LO}}$	$\mathrm{ARE}_{g_{LO},ld}$
20	.05	3	.734	.767	.952	1.72
20	.20	3	-.196	.821	.960	3.10
5	.05	3	1.54	.957	.977	1.08
5	.20	3	.996	.957	.983	1.24

The results so far show that, in several analytical examples, it is possible to achieve reasonably good performance from this simple approximation.

Simulation

To see how well this system might work in practice, some actual physical noise was used to drive the system. The noise was collected underneath the Arctic ice pack, and details may be found in [13]. The noise data is highly nonstationary; a background Gaussian noise is abruptly interrupted with segments of a very heavy tailed noise generated by cracking of the Arctic ice pack. The kurtosis for each consecutive block of 1024 noise samples was calculated, and blocks with a kurtosis exceeding 4 were selected. To get a more nearly stationary noise for driving the system, the data in each block was normalized to unit variance, randomly permuted, and the blocks were then concatenated, thus simulating the output of a physical noise source. This noise was used as the input for the adaptive system. The exponent $\hat{c}$ was estimated from the running estimate of $\hat{P}_T$, which converges to the true tail probability as $N \to \infty$. The scale parameter ν cannot be found via numerical methods, since in this case the simulated system has no knowledge of the true density generating the noise observations. However, the Kiefer-Wolfowitz stochastic approximation method can be used to find the value of ν^* which maximizes $\xi_{\hat{g}}$. The convergence rate towards ν^* is fixed by the particulars of the SA algorithm, and no formal attempts were made to optimize its performance.

Figure 8 shows the running estimate of $\hat{c}$ and ν as a function of sample number. Figure 9 shows the estimated efficacy of $\hat{g}$ for each 1024 sample block, and the cumulative estimate of efficacy starting from the first noise observation. At the end of the simulation, it was assumed that the approximated ZNL was as near optimal as possible. The final ZNL is shown in Figure 10. The efficacy of this suboptimal detector was calculated using the empirical distribution of the entire noise observation set, and it was found to be 1.39. The true distribution of the noise is not known, and therefore it is not possible to calculate the efficacy of the true LO nonlinearity. However, it is possible to conclude that the adaptive system was able to use the noise observations and adapt the detector structure in a constructive way. The ARE of the approximate ZNL relative to the linear detector is given simply by the efficacy in this case, and therefore the adaptive ZNL shows an improved performance over the linear.detector.

5. CONCLUSION

The conclusion to be drawn from this study is that it is possible to implement an adaptive detector nonlinearity using fairly simple techniques. The estimate of tail behavior is quite simple: merely a measurement of the relative number of samples exceeding a specified threshold. This apparently gives enough information about tail behavior of the true noise density, so that even a crude approximation to the true ZNL in the tail regions results in fairly good performance. A more sophisticated estimate of $\hat{c}$ might improve the performance $\hat{g}$. It would be interesting to discover how much additional complexity any resulting performance gain could justify.

Some natural simplifications to this approximation method suggest themselves for investigation. One is to replace the central region polynomial in the ZNL simply by a linear connection between the tail approximants at $\pm x_0$. A further simplification would be to replace the power-law tails with piecewise linear approximations, possibly even a single straight line. Some work done by others [7] in approximation of LO nonlinearities suggest that even very simple approximants of the optimal nonlinearity have the potential to achieve performance which is acceptably near the optimal.

ACKNOWLEDGEMENT

This research is supported by the Office of Naval Research under Contract N00014-81-K-0146 and by the National Science Foundation under Grant ECS-79-18915.

REFERENCES

[1] J. H. Miller and J. B. Thomas, "Detectors for Discrete-Time Signals in Non-Gaussian Noise," *IEEE Trans. Inform. Theory*, vol. IT-18, no. 2, pp. 241-250, March 1972.

[2] P. J. Huber, "A Robust Version of the Probability Ratio Test," *Ann. Math. Statist.*, vol. 36, pp.1753-1758, Dec. 1965.

[3] J. B. Thomas, "Nonparametric Detection," *Proc. IEEE*, vol.58, no. 5, pp.623-631, May 1970.

[4] J. W. Modestino, "Adaptive Detection of Signals in Impulsive Noise Environments," *IEEE Trans. Commun.*, vol. COM-25, no. 9, pp. 1022-1026, Sept. 1977.

[5] S. L. Bernstein, *et al*, "Long-Range Communications at Extremely Low Frequencies," *Proc. IEEE*, vol. 62, no. 3, pp. 292-312, March 1974.

[6] A. D. Spaulding and D. Middleton, "Optimum Reception in an Impulsive Interference Environment - Part I: Coherent Detection," *IEEE Trans. Commun.*, vol COM-25, no. 9, pp. 924-934, Sept. 1977.

[7] J. H. Miller and J. B. Thomas, "Robust Detectors for Signals in Non-Gaussian Noise," *IEEE Trans. Commun.*, vol. COM-25, no. 7, pp. 686-690, July 1977.

[8] A. H. El-Sawy and V. D. Vandelinde, "Robust Detection of Known Signals," *IEEE Trans. Inform. Theory*, vol. IT-23, no. 6, pp. 722-727, Nov. 1977.

[9] F. R. Hampel, "The Influence Curve and its Role in Robust Estimation," *Jour. Amer. Stat. Assoc.*, vol. 69, no. 346, pp. 383-393, June 1974.

[10] R. F. Ingram and R. Houle, "Performance of the Optimum and Several Suboptimum Receivers for Threshold Detection of Known Signals in Additive, White, Non-Gaussian Noise," Technical Report 6339, Naval Underwater Systems Center, New London, CT, Nov. 24, 1980.

[11] D. F. Andrews, *et al*, *Robust Estimates of Location*, Princeton University Press, Princeton, NJ, 1972.

[12] A. D. Watt and E. L. Maxwell, "Measured Statistical Characteristics of VLF Atmospheric Noise," *Proc. Inst. Radio Engineers*, vol. 45, no. 1, pp 55-62, Jan. 1957.

[13] R. F. Dwyer, "Arctic Ambient Noise Statistical Measurement Results and Their Implications to Sonar Performance Improvements," Technical Report 6739, Naval Underwater Systems Center, New London, CT, May 5, 1982.

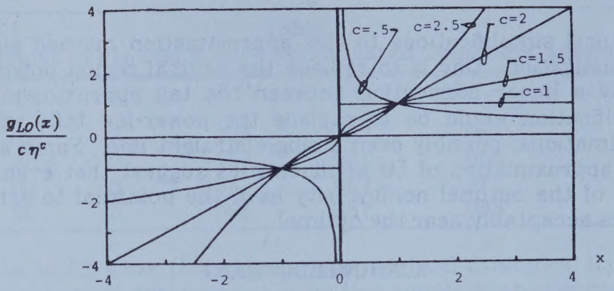


Figure 1. Locally optimal nonlinearities for the generalized Gaussian family.

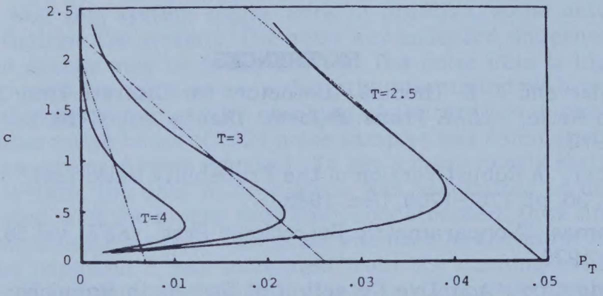


Figure 2. The exact relation $c = h_T(P_T)$ (solid line), and its approximation $\hat{c} = \hat{h}_T(P_T)$ (broken line) for various T.

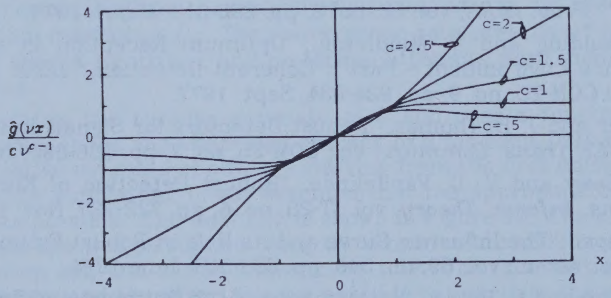


Figure 3a. The ZNL $\hat{g}$ for $x_0 = 3$, $\nu = 1$, various c.

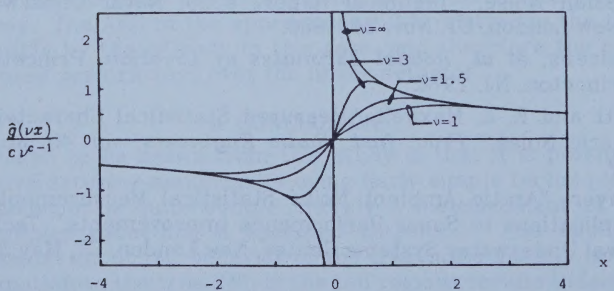


Figure 3b. The ZNL $\hat{g}$ for $x_0 = 3$, various ν, $c = .5$.

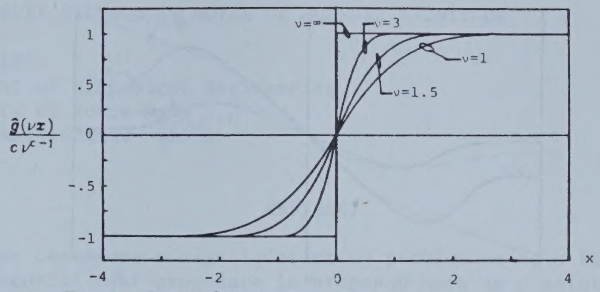


Figure 3c The ZNL $\hat{g}$ for x_0 = 3, various ν, c = 1.

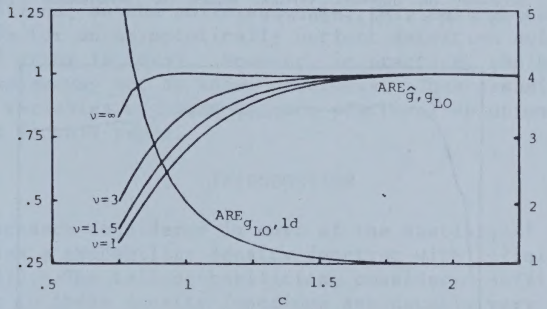


Figure 4. Performance of $\hat{g}$ and g_{LO} versus c for the generalized Gaussian family. $\mathrm{ARE}_{\hat{g},g_{LO}}$ left scale; $\mathrm{ARE}_{g_{LO},ld}$ right scale.

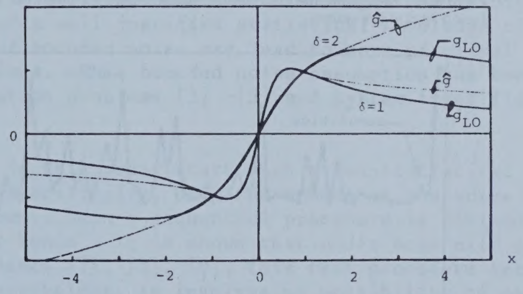


Figure 5. g_{LO} (solid line) and $\hat{g}$ (broken line) for Johnson S_u family, on different vertical scales for comparison. For δ = 1, $\hat{c}$ = .752, ν^* = 3.26, x_0 = 3. For δ = 2, $\hat{c}$ = 1.46, ν^* = 1.88, x_0 = 3.

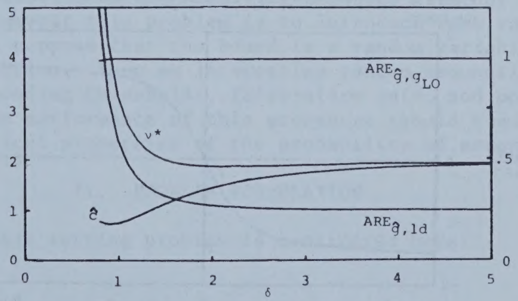


Figure 6. Performance of $\hat{g}$ and g_{LO} versus δ for the Johnson S_u family. $\mathrm{ARE}_{\hat{g},g_{LO}}$, right scale; all others on left scale.

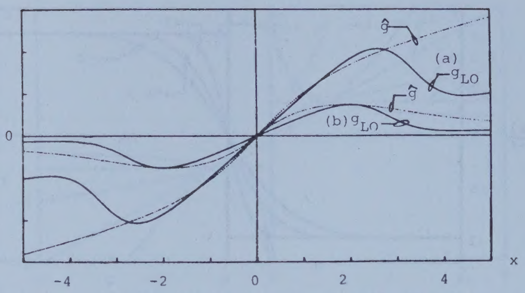


Figure 7. g_{LO} (solid line) and $\hat{g}$ (broken line) for Gaussian-Gaussian mixture, on different vertical scales for comparison. (a) $\varepsilon = .05 \quad \sigma_1^2 = 5$; (b) $\varepsilon = .20 \quad \sigma_1^2 = 20$.

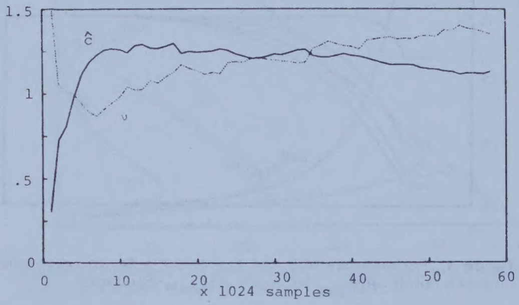


Figure 8. ZNL parameters versus observation number for heavy tailed Arctic ice noise.

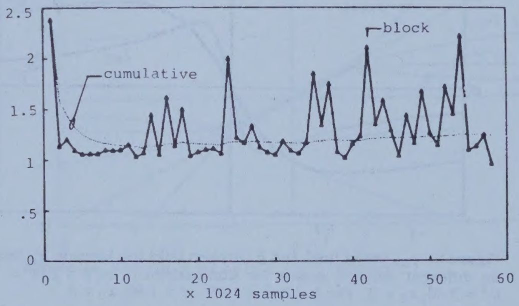


Figure 9. Estimated performance of $\hat{g}$ for the heavy tailed Arctic ice noise.

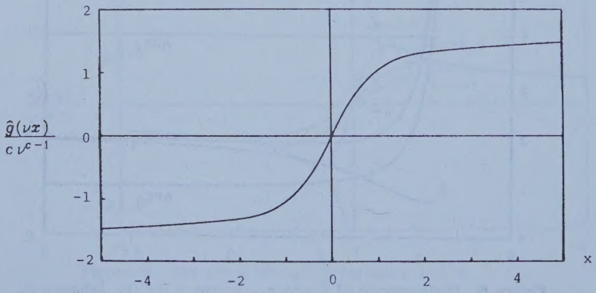


Figure 10. Estimated ZNL at end of Arctic ice noise simulation. $\hat{c} = 1.13$, $\nu^* = 1.35$, $x_0 = 3$.

DETECTION OF SURE SIGNALS IN NOISE OF BOUNDED MAGNITUDE

Y. F. HUANG
Department of Electrical Engineering
University of Notre Dame
Notre Dame, Indiana 46556

ABSTRACT

This paper considers a signal detection problem using a bounded noise model. A sequential test procedure is proposed here as a solution to this detection problem. Due to the non-statistical assumptions; namely, the boundedness assumptions, on the noise sequence, this test procedure leads to a singular solution (or an asymptotically perfect detection solution; i.e. the probability of error is zero). However, in practice, the bound on the magnitude of the noise may not be known precisely. Thus assuming that the bounds are random variables may lead to more practical solutions. This issue also addressed in this paper.

I. INTRODUCTION

The noise processes considered in most of the statistical communication literature often has a probability density function with infinite support (see e.g., [1], [2]). The tail probabilities, considered sufficiently far out, corresponding to these density functions are usually very small and thus are practically neglegible. In practice, due to some environmental limitations, the noise may actually be bounded in magnitude. Furthermore, the availability of a well justified bound on noise magnitude is often more realistic than that of a well justified statistical knowledge of the noise. Thus the assumption of bounded noise may lead to more practical applicability of the solutions. This bounded noise assumption has been employed in considering estimation problems [3] -[5] and system identification problems [6].

The discussions in this paper start with a non-statistical assumption on the input model. Specifically, only the bounds on the noise magnitude are assumed to be known. Then a sequential procedure is devised based only on the known constant bound. It is shown that under some mild statistical assumptions; m-dependence [7], [8], [9]; this test procedure terminates with probability one. Nevertheless, it involves no possibility of making wrong decisions when terminated. The result is inherent from the non-statistical assumptions imposed here. In general, the presumed bound on the noise magnitude may not be precise, thus the resulting solution will not be realistic. One way to circumvent this problem is to introduce some randomness to the bound; namely, to suppose that the bound is a random variable with a known bounded distribution. Then an interesting random sequential test evolves. The corresponding thresholds, false-alarm rate, and power are all random variables. The performance of this procedure should then be determined via the statistical properties of the probability of error.

II. PROBLEM FORMULATION

A binary hypothesis testing problem is considered here:

$$\begin{aligned} H&: x_i = n_i \\ & \qquad\qquad i = 1, 2, \ldots \\ K&: x_i = n_i + s \end{aligned} \tag{1}$$

In contrast to the usual ones, the essential assumption here is that the noise sequence $\{n_i\}$ is bounded in magnitude, namely

$$n_i^2 \leq B^2 \quad \text{for any } i, \text{ where } B > 0 \tag{2}$$

Now, if it is assumed that the signal is a known positive constant, then

$$\text{under H: } n_i^2 \leq B^2 \quad \Longleftrightarrow \quad -B \leq x_i \leq B \tag{3}$$

$$\text{under K: } n_i^2 \leq B^2 \quad \Longleftrightarrow \quad s - B \leq x_i \leq s + B \tag{4}$$

It is clear that $x_i < s-B$ only if H is true and $x_i > B$ only if K is true. Thus, one trivial way to solve the detection problem is to proceed as follows:

1. Obtain the data sequence $\{x_i\}$ sequentially.
2. For any i, if $x_i > B$, stop and decide K. On the other hand, if $x_i < s-B$, stop and decide H.
3. If $s - B < x_i < B$, obtain another observation x_{i+1} and repeat step 2.

Note that if the bound B is known exactly, and if the test procedure terminates, one can expect no possibility of making wrong decisions. In contrast to conventional detection problems, this is a singular problem in the sense that the test procedure leads to a solution of zero false-alarm rate and unity power. This result is inherent from the non-statistical assumption imposed here, namely, that the noise sequence is bounded in magnitude.

The sequential procedure discussed in this paper involves some similarities to the standard sequential probability ratio test (SPRT) [10]. In the SPRT, one assumes that the noise sequence is a sequence of random variables with a known probability distribution. Then the likelihood ratio L is computed and is usually a function of time. Two thresholds T_1 and T_2 are chosen based on the desired values of the false-alarm rate and the power. On the other hand, instead of computing the likelihood ratio each time after making an observation, the sequential procedure discussed in this paper simply compares the observed data x_i to the two thresholds B and s-B to make decisions. Thus it requires much less information about the noise and involves less computational complexity. Furthermore, it can be shown that, with the introduction of some mild statistical assumptions, this sequential test procedure will terminate with probability one.

Suppose that the observed data x_i lies in the interval $[s-B, B]$ with a probability p_i, where $0 < p_i < 1$ for all i. Let the observation sequence $\{X_i\}$ be where an m-dependent sequence, where $m < \infty$. Then, there must exist an independent subsequence of $\{X_i\}$, $\{X_{a_j}\}$, where $|a_j - a_k| \geq m$ for $j \neq k$. At the k-th observation, an upper bound, ρ_k, for the probability that this test procedure has not terminated is given by

$$\rho_k = \prod_{j=1}^{r} p_{a_j} \leq p_{\max}^{r}$$

where

$$p_{max} = \max_{1 \leq a_j \leq r} p_{a_j}$$

and r is the maximum integer such that $rm \leq k$. Obviously, $\lim_{k \to \infty} \rho_k = 0$.

The above argument leads to the following lemma.

Lemma : Let $\{X_i\}$ be an m-dependent sequence. Let p_i = Prob. $\{x_i \varepsilon [s-B, B]\}$. Then, if $0 < p_i < 1$ for all i, the sequential test procedure discussed in this paper terminates with probability one.

In general, the cost of the test is dependent on the number of samples required to terminate it. Thus, one can usually consider the asymptotic sample number (ASN) as a performance index, if the procedure terminates with probability one. Now, assume that the observation sequence is independent and identically distributed.

(a) Suppose that the noise sequence $\{N_i\}$ is uniformly distributed between [-B, B]; then, under H:

$$\text{Prob. } \{(x: -B \leq x \leq s - B)\} = \frac{s}{2B}$$

Thus the expected value of the sample number M is, by the strong law of large numbers,

$$E_H(M) = \frac{2B}{s} \rightarrow \text{ASN}$$

Similarly, under K:

$$E_K(M) = \frac{2B}{s} \rightarrow \text{ASN}$$

One can easily see that the two ASN's may not be equivalent under both hypothesis if one allows some asymmetry assumption. Furthermore, it should be noted that, if B>>s, then the probability that the observation will lie in [-B, s-B] will be very small and thus the ASN will be very large. On the other hand, if B and s are comparable, one can expect a rapid termination of the test.

(b) If the noise sequence $\{N_i\}$ has a truncated Gaussian distribution, i.e. if it has a density function given by

$$f_N(n) \begin{cases} = \dfrac{1}{\sqrt{2\pi\sigma^2}\,(1-2\Phi(-B/\sigma))} \exp^{-n^2/2\sigma^2} & \text{if } |n| \leq B \\ = 0 & \text{otherwise} \end{cases}$$

Then it can be shown similarly that

$$\text{ASN} = \frac{(1 - 2\Phi(-B/\sigma))}{[\Phi(\frac{s - B}{\sigma}) - \Phi(-B/\sigma)]}$$

under either H or K.

III. CONSTANT KNOWN SIGNALS IN NOISE OF RANDOM BOUNDS

In the previous section, the problem of detecting a known signal in a magnitude bounded noise has been discussed. A sequential test procedure was proposed which yields a singular solution. However, there are some obvious problems associated with this result. The presumed bound on the noise magnitude may not be tight, and thus the number of samples required to terminate the test procedure may be unnecessarily large. Furthermore, in practice, the bound may not be known exactly, and could even be assumed smaller than it actually is. Therefore, to make the problem more realistic, one may assume that the presumed bound is a random variable with a known distribution function. Then, clearly, the two thresholds will also be random variables. Thus the sequential procedure becomes a random test.

For simplicity of discussion, one may suppose that the presumed bound B is a random variable uniformly distributed between $[r-\varepsilon, r+\varepsilon]$, where r is the actual bound of the noise sequence $\{n_i\}$. A realization of B will be denoted by b. Then, assuming that the signal is a known positive constant, the two thresholds s-B and B will be random variables uniformly distributed between $[s-r-\varepsilon, s-r+\varepsilon]$ and $[r-\varepsilon, r+\varepsilon]$, respectively. It can be seen that a false-alarm (type I error) occurs when $b > r$ and $x_i > r$. Hence the probability of false-alarm (false-alarm rate) is given by

$$\alpha(b) = \text{Prob. } \{b>r,\ x_i>r\}$$

$$= \text{Prob. } \{x_i>r \mid b>r\} \text{ Prob. } \{b>r\} \tag{5}$$

Moreover, a miss (type II error) occurs when $b > r$ and $x_i < s-r$. Hence the probability of miss is

$$1-\beta(b) = \text{Prob. } \{b>r,\ x_i<s-r\}$$

$$= \text{Prob. } \{x_i<s-r \mid b>r\} \text{ Prob. } \{b>r\} \tag{6}$$

Thus the power $\beta(b)$ is given by

$$\beta(b) = 1 - \text{Prob. } \{x_i<s-r \mid b>r\} \text{ Prob. } \{b>r\} \tag{7}$$

It is seen in Eq. (5) and Eq. (6) that both the false-alarm rate α and the power β are functions of a random variable B and thus are also random variables. Therefore, this test procedure can be regarded as a random sequential procedure as its thresholds are random and also its performance. Properties of this test procedure will clearly depend on the statistical distributions of the false-alarm rate and the power. Two particular examples are given in the sequel where some statistical assumptions on the noise sequence are imposed.

A. Uniformly Distributed Random Noise

Suppose that for any presumed bound b, the noise sequence $\{N_i\}$ is uniformly distributed between $[-b, b]$. Thus, under H, $\{X_i\}$ is uniformly distributed on $[-b, b]$ and, under K, $\{X_i\}$ is uniformly distributed on $[s-b, s+b]$. It may be interesting to see that the probability density function of X_i under H is given by

$$\left(= \frac{1}{4\varepsilon} \ln\left(\frac{r+\varepsilon}{x} \right) \qquad \text{if } r + \varepsilon \geq x > r - \varepsilon \right.$$

$$p_X(x) \begin{cases} = \dfrac{1}{4\varepsilon} \ln\left(\dfrac{r+\varepsilon}{r-\varepsilon}\right) & \text{if } r-\varepsilon \geq x > -r+\varepsilon \\ = \dfrac{1}{4\varepsilon} \ln\left(\dfrac{r+\varepsilon}{-x}\right) & \text{if } -r+\varepsilon \geq x > -r-\varepsilon \\ = 0 & \text{otherwise} \end{cases} \tag{8}$$

The probability density function of X_i under K is simply a shift to the right of Eq. (8). As discussed previously, a false-alarm occurs when $b > r$ and $x_i > r$, while a miss occurs when $b > r$ and $x_i < s-r$. The false-alarm rate and the power can then be evaluated via Eqs. (5) and (7), respectively. From Eq. (5),

$$\begin{aligned} \alpha(b) &= \text{Prob. } \{x_i > r \mid b > r\} \text{ Prob. } \{b > r\} \\ &= \frac{b-r}{2b} \cdot \frac{1}{2} = \frac{1}{4} - \frac{r}{4b} \end{aligned} \tag{9}$$

Similarly, from Eq. (7),

$$\begin{aligned} \beta(b) &= \text{Prob. } \{x_i < s-r \mid b > r\} \text{ Prob. } \{b > r\} \\ &= \frac{3}{4} + \frac{r}{4b} \end{aligned} \tag{10}$$

Now that both the false-alarm rate and the power are now random variables [Eqs. (9) and (10)]. To evaluate the performance of this test procedure, it is essential to consider the statistical properties of $\alpha(b)$ and $\beta(b)$. The probability density function of $\alpha(b)$ is given by

$$f_A(\alpha) \begin{cases} = \dfrac{2r}{\varepsilon(1-4\alpha)^2} & \text{if } \dfrac{-\varepsilon}{4(r-\varepsilon)} \leq \alpha \leq \dfrac{\varepsilon}{4(r+\varepsilon)} \\ = 0 & \text{otherwise} \end{cases} \tag{11}$$

and that of $\beta(b)$ by

$$f_B(\beta) \begin{cases} = \dfrac{2r}{\varepsilon(1-4\beta)^2} & \text{if } \dfrac{3}{4} + \dfrac{r}{4(r+\varepsilon)} \leq \beta \leq \dfrac{3}{4} + \dfrac{r}{4(r-\varepsilon)} \\ = 0 & \text{otherwise} \end{cases} \tag{12}$$

Moreover, the first two statistical moments of the false-alarm rate and the power are given by

$$E[\alpha(b)] = \frac{1}{8} - \frac{1}{8t} \ln(1+t) \tag{13}$$

where

$$t = \varepsilon/r$$

$$E[\beta(b)] = \frac{7}{8} + \frac{1}{8t} \ln(1+t) \tag{14}$$

and

$$\mathrm{var}[\alpha(b)] = \mathrm{var}[\beta(b)]$$
$$= \frac{1}{64} + \frac{1}{32(1+t)} - \frac{1}{32t} \ln(1+t) - \frac{1}{64t^2} [\ln(1+t)]^2 \qquad (15)$$

Eqs. (11) and (12) show that both α and β have bounded distributions with an algebraic tail behavior. In practice, one may be interested in cases where $t \ll 1$, then it can be shown that $E[\alpha(b)]$ and $E[\beta(b)]$ can be approximated by

$$E[\alpha(b)] \approx \frac{t}{16}$$

$$E[\beta(b)] \approx 1 - \frac{1}{16} t$$

and

$$\mathrm{Var}[\alpha(b)] = \mathrm{Var}[\beta(b)] \approx \frac{1}{32} t^2$$

Hence if $t \ll 1$, the means of both α and β are monotone linear functions of t while the variance is a quadratic frunction of t. This result suggests a reasonable conclusion that, the smaller t is, the smaller is the probability of error involved in this test procedure.

B. Random Noise with Truncated Gaussian Distribution

In applications such as radar or sonar, the noise density function often exhibits a Gaussian-shaped central part, although the magnitude of the noise may be bounded. The probability density function of a zero mean random variable with a truncated Gaussian distribution is given by

$$f_X(x) \begin{cases} = \dfrac{C(b)}{\sqrt{2\pi}\sigma} \exp^{-x^2/2\sigma^2} & \text{if } |x| \le b \\ = 0 & \text{Otherwise} \end{cases} \qquad (16)$$

where C(b) is a normalization constant and is given by

$$C(b) = [\Phi(\frac{b}{\sigma}) - \Phi(\frac{-b}{\sigma})]^{-1}$$

Again, the bound b is assumed to be a random variable uniformly distributed between $[r-\varepsilon, r+\varepsilon]$. Then, by Eqs. (5) and (7), the false-alarm rate and power are given by

$$\alpha(b) = \text{Prob. } \{x_i > r,\ b > r\}$$
$$= \frac{C(b)}{2} [\Phi(\frac{b}{\sigma}) - \Phi(\frac{r}{\sigma})]$$

and

$$\beta(b) = 1 - \frac{C(b)}{2} [\Phi(\frac{s-r}{\sigma}) - \Phi(\frac{s-b}{\sigma})]$$

Because of the functional forms involved in the density function, Eq. (16), the statistical moments of α and β are hardly expressible in a simple form. However, curves depicting them as functions of t can be obtained by using numerical methods as shown in Figs. 1-3. These curves are obtained for t varying from 0.001 to 0.1. Also, the value of r is set at 4σ, as the tail probability of a Gaussian distribution beyond this point is sufficiently small. A smooth monotone nondecreasing relation between t and the first two moments of the probability of error can be seen from these curves. This is compatible with the intuition that the more accurate the presumed bound is, the less probability of error will result from this test procedure.

To investigate in more detail the sequential test procedure here, one may be interested in comparing its performance to that of the standard SPRT. It is obvious that both error probabilities converge to zero as the bound on the noise distribution goes to infinity, namely

$$\lim_{r\to\infty} \alpha(b) = 0 \qquad \text{almost surely}$$

and

$$\lim_{r\to\infty} \beta(b) = 1 \qquad \text{almost surely}$$

Unfortunately, the ASN will be infinite in the limit. A similar result can be expected from the standard SPRT. Thus, to compare these two test procedures, one has to consider rate of convergence of the error probabilities. In [10], it is shown that the ASN of the SPRT under H and under K are given respectively by

$$E_H(M) = \frac{(1-\alpha)\log T_1 + \alpha \log T_2}{E_H(z)}$$

and

$$E_K(M) = \frac{\beta \log T_1 + (1-\beta)\log T_2}{E_K(z)}$$

where z is the test statistic and T_1 and T_2 are two thresholds given by

$$T_1 = \beta / (1-\alpha) \qquad \text{and} \qquad T_2 = (1-\beta)/\alpha$$

Thus one can see that for the same α and the same β, the ASN of the SPRT is less than that of the sequential procedure here. On the other hand, a complete knowledge on the noise density function required by the SPRT and the computational simplicity involved in the proposed sequential procedure should also be taken into account in making comparisons.

IV. CONCLUSIONS

In this paper, detection of a constant signal in bounded noise is considered. When the bound is a known constant, the solution is singular, namely, both the false-alarm rate and the probability of miss are zero. To make the problem more realistic, we considered noise with a random bound on the magnitude ; then a random test is obtained. Some statistical properties of the performance parameters, e.g. the false-alarm rate and the power, are obtained for two examples with presumed assumptions on the distributions of the noise and the bound. As can be expected, the probability of error of the test is dependent on the tail property of the noise distribution and, of course, the accuracy of the presumed bound on the noise magnitude. In contrast to the SPRT, the test procedure discussed in this paper requires very little statistical information on the noise and involves less computational complexity, although it may be terminated with a larger sample size.

REFERENCES

[1] J. Marcum, "Statistical Theory of Target Detection by Pulsed Radar," IRE Trans. on Information Theory, Vol. IT-6, pp. 59-267, 1960.

[2] J.H. Miller and J.B. Thomas, "Detectors for Discrete-Time Signals in Non-Gaussian Noise," IEEE Trans. Inform. Theory, Vol. IT-18, pp. 241-250, 1972.

[3] F.C. Schweppe, "Recursive State Estimation: Unknown but Bounded Errors and System Inputs," IEEE Trans. Automat. Contr., Vol. AC-13, pp. 22-28, 1968.

[4] F.M. Schlaepfer and F.C. Schweppe, "Continuous-Time State Estimation Under Disturbances Bonded by Convex Sets," IEEE Trans. on Automat. Contr., Vol. AC-15, pp. 197-205, 1970.

[5] M. Milanese and G. Belforte, "Estimation Theory and Uncertainty Intervals Evaluation in Presence of Unknown But Bounded Errors: Linear Families of Models and Estimators," IEEE Trans. on Automat. Contr., Vol. AC-27, pp. 408-414, 1982.

[6] E. Fogel and Y.F. Huang, "On The Value of Information in System Identification-Bounded Noise Case," Automatica, Vol. 18, pp. 229-238, 1982.

[7] P. Billingsley, Probability and Measure, New York: John Wiley & Sons, Inc., 1979.

[8] H.V. Poor, "Signal Detection in the Presence of Weakly Dependent Noise -Part I: Optimum Detection," IEEE Trans. Inform. Theory, Vol. IT-28, pp. 735-744, 1982.

[9] H.V. Poor, "Signal Detection in the Presence of Weakly Dependent Noise -Part II: Robust Detection," IEEE Trans. Inform. Theory, Vol. IT-28 pp. 744-752, 1982.

[10] A. Wald, Sequential Analysis, New York: Dover, 1973.

ACKNOWLEDGEMENT

The author wishes to express his sincere appreciations to Professor John B. Thomas for his helpful discussions and many valuable comments on this work. This research was supported by the U.S. Army under Grant DAAG29-82-K-0095 and by the National Science Foundation under Grant ECS-79-18915.

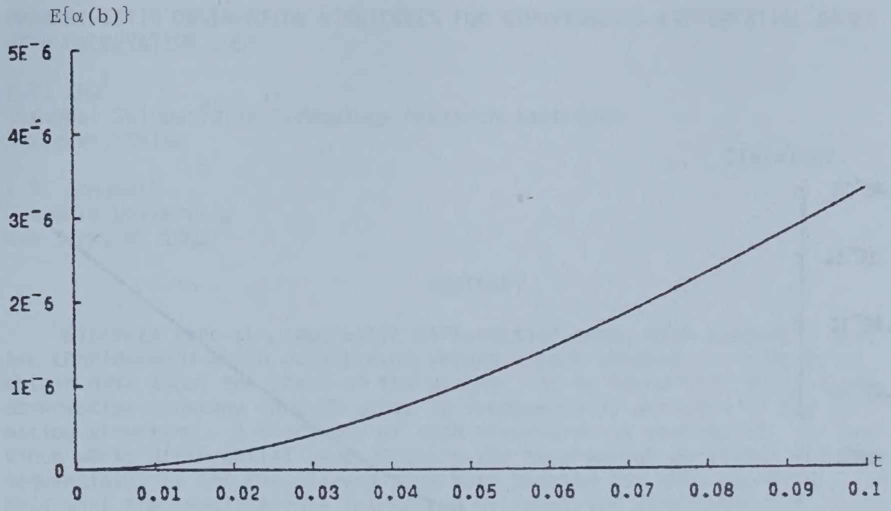


Fig. 1 Mean of α

The noise has a truncated Gaussian distribution.

$$t \triangleq \frac{\varepsilon}{\gamma}$$

$$\gamma = 4\sigma$$

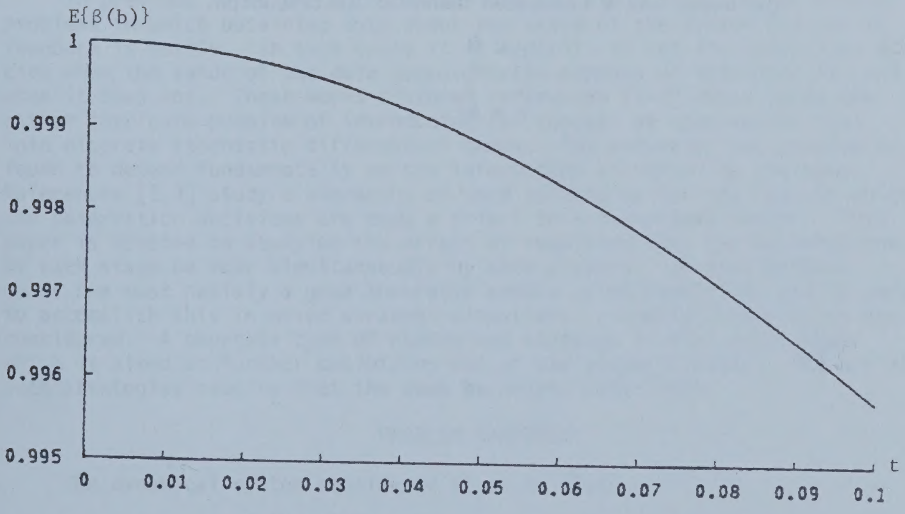


Fig. 2 Mean of β

The noise has a truncated Gaussian distribution.

$$t \triangleq \frac{\varepsilon}{\gamma}$$

$$\gamma = 4\sigma, \ \frac{s^2}{\sigma^2} = 6.25$$

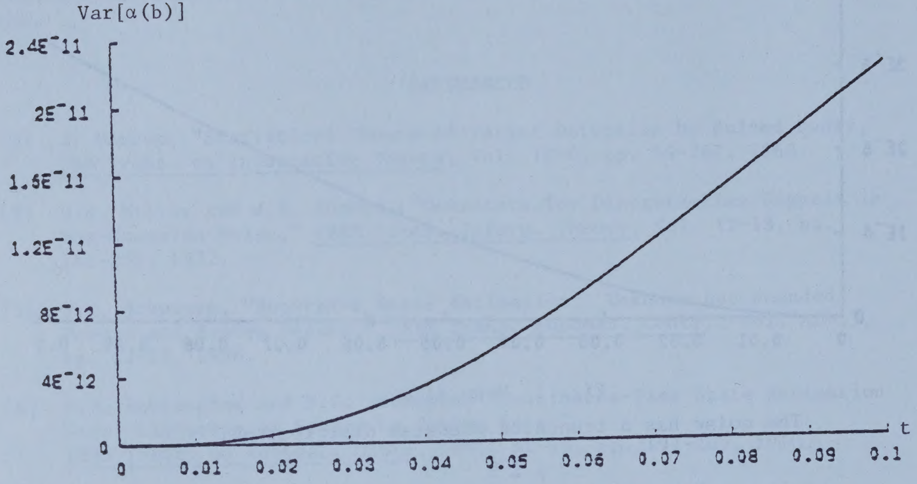


Fig. 3 Variance of α

The noise has a truncated Gaussian distribution.

$$t \triangleq \frac{\varepsilon}{\gamma}$$

$$\gamma = 4\sigma$$

PROBABILISTIC OBSERVATION STRATEGIES FOR SIMULTANEOUS DIFFERENTIAL GAMES WITH OBSERVATION COST

Z.T. Zhu[1]
Shanghai Shipbuilding Technology Research Institute
Shanghai, China

R.W. Longman[2]
Columbia University
New York, NY 10027

ABSTRACT

Discrete zero-sum stochastic differential games with quadratic cost are considered in which each player incurs a cost whenever he chooses to obtain data about the state of the system. It is found that the optimal observation strategy in such games is fundamentally affected by the information structure. A hierarchy of such structures is considered. In previous works differential games in which the observation decisions are made sequentially by the two players have been treated for this hierarchy. In this work the complications introduced by requiring simultaneous decision making at each stage are considered. Two types of randomized or probabilistic strategies are considered. The first is designed to avoid mixed strategy situations, and the second is used by a player in order to further optimize his cost in an ensemble average sense over many games.

INTRODUCTION

In previous works the authors have studied stochastic optimal control problems in which obtaining data about the state of the system for use in feedback is costly. In such cases it is logical to let the controller decide when the value of the data outweights the expense of obtaining it, and when it does not. These works fostered references [1-3] which study the rather intricate problem of introducing the concept of observation cost into discrete stochastic differential games. The nature of the problem is found to depend fundamentally on the information structure of the game. References [2,3] study a hierarchy of such structures for the case in which the observation decisions are made a priori in a sequential manner. This paper is devoted to studying the effect of requiring that the observations at each stage be made simultaneously by both players. In simultaneous games one must satisfy a game theoretic saddle point condition, and in order to accomplish this in mixed strategy situations, randomized strategies are considered. A separate type of randomized strategy is also introduced which is aimed at further optimizing one of the player's costs. The use of such strategies require that the game be played many times.

PROBLEM STATEMENT

The dynamical system considered is as in [2,3]

$$X_{k+1} = A_K X_K + B_k^p u_k^p + B_k^e u_k^e + \xi_K \qquad 0 \leq K \leq N-1$$

---

[1] Work performed while this author was a Visiting Scholar at Columbia Univ.

[2] Professor of Mechanical Engineering.

$$
\begin{aligned}
y_k^p &= C_k^p x_k + \eta_k^p \\
y_k^e &= C_k^e x_k + \eta_k^e \\
w_k^p &= i_k^p y_k^p \\
w_k^e &= i_k^e y_k^e
\end{aligned}
\tag{1}
$$

Where u_k^p and u_k^e are the pursuer's and evader's controls, w_k^p and w_k^e are their corresponding data vectors, and the observation indicators i_k^p and i_k^e can assume only the values of 0 or 1 depending on whether an observation is purchased by the player. Variables ξ_k, η_k^p and η_k^e are uncorrelated, zero-mean, white gaussian noises with covariances $\mathcal{H}_k$, T_k^p and T_k^e respectively, and the initial condition x(o) is gaussian with mean $\overline{x}_o$ and covariance and is uncorrelated with the plant and measurement noises.

The pursuer wishes to minimize, and the evader to maximize, by choice of their respective controls and observation indicators, the quantity

$$
J = E\left\{ \sum_{k=o}^{N-1} \left(x_k^T Q_k x_k + u_k^{pT} R_k^p u_k^p - u_k^{eT} R_k^e u_k^e + i_k^p \Box_k^p - i_k^e \Box_k^e \right) + x_N^T Q_N x_N \right\} \tag{2}
$$

based on the information available to each at the time the decisions must be made. It is assumed that $Q_N > o$, $R_k^p > o$, $R_k^e > o$, and $Q_k > o$. The $\Box_k^p \geq o$ and $\Box_k^e \geq o$ are scalar observation costs.

Capitol letters I_k^e, I_k^p, W_k^e, W_k^p, U_k^e, U_k^p are used to denote the set of variables i_j^e, i_j^p, w_j^e, w_j^p, u_j^e and u_j^p fot j = 0, 1, 2, . . ., k. It is assumed that both players determine their observation sequences (I_{N-1}^e, I_{N-1}^p) a priori simultaneously step by step and that the sequence of events at stage K during the game is given as follows: 1) both players receive their data vectors w_k^e and w_k^p simultaneously 2) both players choose their controls u_k^e and u_k^p simultaneously, 3) the system generates x_{k+1}. The hierarchy of information structures considered, numbered IS 1 through IS 7 specifies what information is available to each player after the opponent has taken some action. Table 1 identifies for stage k what kind of information is known by each player concerning his opponent.

MIXED STRATEGIES IN OBSERVATION DECISIONS

The reader is referred to Reference [2], which is notationally consistent with the present work, for details and background for various parts of this discussion. For IS 1 the optimal observation strategy is found to satisfy

$$
\delta_k(i_k^e, i_k^p) = \sum_{i=o}^{m-1} tr\left[P_{k+i}(I_{k+i}^e, I_{k+i}^p) \alpha_{m-i} \right] - i_k^e \Box_k^e + i_k^p \Box_k^p \tag{3}
$$

$$
\begin{aligned}
\alpha_m = {} & A_k^T \Phi_{m-1} B_k^p \left[R_k^p + B_k^{pT} \Phi_{m-1} B_k^p \right]^{-1} B_k^{pT} \Phi_{m-1} A_k \\
& - A_k^T \left[\Phi_{m-1}^{-1} + B_k^p R_k^{p-1} B_k^{pT} \right]^{-1} B_k^e \left[R_k^e - B_k^{eT} \left(\Phi_{m-1}^{-1} + B_k^p R_k^{p-1} B_k^{pT} \right)^{-1} B_k^e \right]^{-1} \\
& \qquad B_k^{eT} \left[\Phi_{m-1}^{-1} + B_k^p R_k^{p-1} B_k^{pT} \right]^{-1} A_k
\end{aligned}
$$

$$
\text{where } \Phi_m = Q_k + A_k^T \left[\Phi_m^{-1} + B_k^p R_k^{p-1} B_k^{pT} - B_k^e R_k^{e-1} B_k^{eT} \right]^{-1} A_k
$$

$$
\Phi_o = Q_N
$$

$$
m = N - K
$$

For simultaneous choices of observation strategies one must require that the saddle point condition $\max_{i_k^e} \min_{i_k^p} \delta_k = \min_{i_k^p} \max_{i_k^e} \delta_k$ be satisfied, or equivalently that $\delta_k(i_k^e, \hat{i}_k^p) \leq \delta_k(\hat{i}_k^e, \hat{i}_k^p) \leq \delta_k(\hat{i}_k^e, i_k^p)$ for all k, with carrot denoting optimal. Two situations in which such a condition is satisfied by the solution obtained to the sequential problem are given in two theorems presented in reference [2]:

Theorem 1: If the matrix α_m is positive definite or positive semidefinite for all m = 1, 2, . . ., N, or it is always negative definite or negative semidefinite for these m, and if $[R_k^e - B_k^{eT} \Phi_{m-1} B_k^e \;] > 0$ is satisfied, then any optimal observation strategy for the sequential problem is also an optimal observation strategy for the simultaneous problem.

Theorem 2: If the measurement noises $\eta_k^e = 0$ and $\eta_k^p = 0$, and C_k^e and C_k^p are invertible for all k = 0, 1, . . ., N-1, and is satisfied, then any optimal observation sequence for the sequential problem is also optimal for the simultaneous problem.

Mixed strategies can easily occur in IS 1, but the following example shows that they can occur more frequently in Class II information structures.

A Single Stage Example

Consider the following system and observation equation with a sequential observation criterion, which can be considered as an example of IS 2, 3 and 4:

$$x_1 = x_0 + u_0^p + u_0^e + \xi_0$$

$$w_0^e = i_0^e x_0$$

$$w_0^p = i_0^p x_0$$

$$J = \max_{i_0^e} E\left\{ \min_{i_0^p} E\left\{ \max_{u_0^e} E\left\{ \min_{u_0^p} E\left[x_1^2 + u_0^{p\,2} - c\,u_0^{e\,2} + i_0^p \square_0^p - i_0^e \square_0^e \right] \middle| i_0^e, i_0^p, w_0^p \right\} \middle| i_0^e, i_0^p, w_0^e \right\} \middle| i_0^e \right\} \tag{4}$$

where it will be assumed that $C > 1$, ξ_0 is zero mean with unit variance, and x_0 has mean $\bar{x}_0$ and variance σ^2 known to both players. Note that the measurements are perfect noiseless measurements.

It is also assumed that u_0^e is of the form $u_0^e = a_0 + a_1 i_0^e x_0$. Substitute this together with x_1 above into the innermost expectation in J. Then differentiation with respect to u_0^p gives $u_0^p = -[(1 + a_1 i_0^e)\hat{x}_0 + a_0]/2$ where $\hat{x}_0 = E x_0 \mid i_0^e, i_0^p, w_0^p$. Then the maximization over u_0^e becomes a maximization over a_0 and a_1, and can be performed for all four choices of i_0^e, i_0^p to obtain the optimal controls given in Table 2. Note that if one of the players makes no measurement, his control law is not influenced by whetehr his opponent makes measurement. Also, the $(i_0^e, i_0^p) = (1,1)$ control laws are the same as those for (0,0) with the value

TABLE 2. The Optimal Controls for the Single Stage Example

(i_0^e, i_0^p)	u_0^e	u_0^p
(0,0)	$\bar{x}_0/(2c-1)$	$-c\bar{x}_0/(2c-1)$
(1,0)	$-c\bar{x}_0/(2c-1)(c-1) + x_0/(c-1)$	$-c\bar{x}_0/(2c-1)$
(0,1)	$\bar{x}_0/(2c-1)$	$-\bar{x}_0/2(2c-1) - x_0/2$
(1,1)	$x_0/(2c-1)$	$-c\,x_0/(2c-1)$

of the measurement x_0 substituted for the a priori estimate $\bar{x}_0$.

Now to determine the optimal observation strategies, evaluate the expectation in J which is minimized in the choice of i_0^p and call it $g(i_0^e, i_0^p)$.

Then for each choice of (i_0^e, i_0^p) we have

$$\begin{aligned} g(0,0) &= G + \sigma^2 \\ g(1,0) &= G + [c/(c-1)]\sigma^2 - \Box_0^e \\ g(0,1) &= G + \sigma^2/2 + \Box_0^p \\ g(1,1) &= G + [c/(2c-1)]\sigma^2 + \Box_0^p - \Box_0^e \end{aligned} \tag{5}$$

$$G = 1 + c\bar{x}_0^2/(2c-1)$$

In the sequential game represented by (4) the above g(i_0^e, i_0^p) takes the place of the $\delta_K(i_0^e, i_0^p)$, and is to be optimized to complete the solution to this example in the case of information structure IS 4. Note that although in (4) it was assumed that the evader decides the value of i_0^e first and then announces it to the pursuer before i_0^p is picked, this announcement does not influence the game in the sense that the pursuer could have calculated .

The effect of changing the information structure to IS 2 or IS 3 while maintaining the sequential nature of the problem, can be seen easily. For IS 2, the pursuer knows the evader's perfect measurement which means that the optimal control and cost g for (i_0^e, i_0^p) = (1,0) must now be the same as for (1,1) although the observation costs 0 will not be present. This means that the g(1,0) of equation (5) must be changed to

$$g(1,0) = G + [c/(2c-1)]\sigma^2 - \Box_0^e \tag{6}$$

Similarly, for IS 3 where the evader knows the pursuer's perfect measurement (i_0^e, i_0^p) = (0,1) will be the same as (1,1) except for the observation cost, and g(0,1) in (5) must be changed to

$$g(0,1) = G + [c/(2c-1)]\sigma^2 + \Box_0^p \tag{7}$$

In each case, for the sequential problem the resulting g matrix is used to pick the observation indicators following the max min rules.

Now consider what changes occur when the example is modified to be a simultaneous problem. In order to have a solution it is necessary to have certain saddle point conditions satisfied for the choice of the controls which require that C > 1 [2]. Also, if the observation indicators for each player are to be chosen simultaneously, then the solution should have the property that

$$\min_{i_0^p} \max_{i_0^e} g(i_0^e, i_0^p) = \max_{i_0^e} \min_{i_0^p} g(i_0^e, i_0^p)$$

The decision matrix g(i_0^e, i_0^p) for IS 4 will fail to satisfy this condition if one of the diagonals dominates. One can show that g(0,0) and g(1,1) both being larger than g(1,0) and g(0,1) produces a contradiction, but the other diagonal can dominate and produce a mixed strategy provided the observation costs 0 and 0 simultaneously satisfy

$$1/2 < \Box_0^p/\sigma^2 < c^2/[(c-1)(2c-1)]$$

$$1/[2(2c-1)] < \Box_0^e/\sigma^2 < 1/(c-1)$$

For information structure IS 2 mixed strategies occur when $\Box_0^p < \sigma^2/2$ and $\Box_0^e < \sigma^2/[2(2c-1)]$, and for IS 3 when $(c-1)/(2c-1) < \Box_0^p/\sigma^2 < c/[(c-1)(2c-1)]$ and $\Box_0^e < \sigma^2/(c-1)$.

This example represents a very simple problem involving only a scalar control and scalar state, including only one time step, and involving perfect measurements only. It has been found that even in such a simple problem, and even if only one of the players fails to know his opponent's measurement, it is possible to obtain a mixed strategy situation. This is in contrast to IS 1 for which either Theorem 1 or Theorem 2 establishes that no mixed strategy can occur.

PROBABILISTIC OBSERVATION STRATEGIES FOR CLASSES I AND II

For the simultaneous game in IS 1 the choice of the observation indicators, which can be made a priori, can result in a mixed strategy at some or all stages of the game. If the game is to be played many times, this difficulty can be alleviated by randomizing the strategy, in which case the saddle point condition is satisfied, and the associated strategy is optimal, relative to an ensemble average of these games. In this section we present a method of generating the optimal probability of making an observation at each stage, for each player.

Consider stage k. Each player knows what choices he has made in the past for observation indicators, and for IS 1 (as well as IS 2-4) he also knows his opponent's past observation indicators, i.e. I_{k-1}^{e} and I_{k-1}^{p} are known to both players. Each player wishes to determine an optimal probability μ_k^e or μ_k^p for making an observation at stage k, and the resulting optimal probabilities have the dependence $\mu_k^e(I_{k-1}^{e}, I_{k-1}^{p})$, $\mu_k^p(I_{k-1}^{e}, I_{k-1}^{p})$.

The choice of μ_k^e and μ_k^p is facilitated by knowledge of the optimal ensemble average total cost conditioned on use of given I_k^e and I_k^p. We denote this by $\hat{g}_{k+1}(i_k^e, i_k^p, I_{k-1}^{e/p}, \hat{M}^{e/p}_{k+1,N-1})$ where the last argument represents the set of optimal μ_j^e and μ_j^p (i.e. $\hat{\mu}_j^e, \hat{\mu}_j^p$) for j = k+1, k+2, . . ., N-1, and superscript e/p is used for compactness of notation to indicate both the associated sets for the evader and the pursuer. Then the total expected cost using I_{k-1}^{e}, I_{k-1}^{p} for step 0 to k-1, using any μ_j^e and μ_j^p, and using the optimal $\hat{\mu}_j^e$, $\hat{\mu}_j^p$ for stages j = k+1, . . ., N-1 is

$$\begin{aligned} g_k(\mu_k^e, \mu_k^p, I_{k-1}^{e/p}, \hat{M}^{e/p}_{k+1,N-1}) = {} & (1-\mu_k^e)(1-\mu_k^p) \quad \hat{g}_{k+1}(0, 0, I_{k-1}^{e/p}, \hat{M}^{e/p}_{k+1,N-1}) \\ & + \mu_k^e(1-\mu_k^p) \quad \hat{g}_{k+1}(1, 0, I_{k-1}^{e/p}, \hat{M}^{e/p}_{k+1,N-1}) \\ & + (1-\mu_k^e)\mu_k^p \quad \hat{g}_{k+1}(0, 1, I_{k-1}^{e/p}, \hat{M}^{e/p}_{k+1,N-1}) \\ & + \mu_k^e \mu_k^p \quad \hat{g}_{k+1}(1, 1, I_{k-1}^{e/p}, \hat{M}^{e/p}_{k+1,N-1}) \end{aligned} \tag{8}$$

Note that at stage k = N, $\hat{g}_N(i_{N-1}^e, i_{N-1}^p, I_{N-2}^{e/p}, -) = J(I_{N-1}^e, I_{N-1}^p)$, the total cost associated with the given observation sequences. This must be calculated for each possible sequence pair. It can be obtained recursively as in [2] and denoted $f_N(I_{N-1}^e, I_{N-1}^p, -, -)$, after checking for satisfaction of the saddle point condition associated with simultaneous control decisions. Note that in IS 1 only a few of the terms in the cost-to-go from stage k are dependent on the observation indicators, and this allows one to substitute for $J_0(I_{N-1}^e, I_{N-1}^p)$ the simpler expression related to δ_k

$$\delta(I_{N-1}^e, I_{N-1}^p) = \sum_{k=0}^{N-1} \left\{ \mathrm{tr}\left[P_k(I_{N-1}^e, I_{N-1}^p)\, \alpha_{N-k} \right] - i_k^e \Box_k^e + i_k^p \Box_k^p \right\} \tag{9}$$

Knowledge of $\hat{g}_N$ allows us to progress backwards from stage N-1 to 0 using equation (8) and optimizing the choice of $\mu_k^e(I_{k-1}^e, I_{k-1}^p)$ and $\mu_k^p(I_{k-1}^e, I_{k-1}^p)$ at each step. For notational compactness we suppress the third and fourth arguments on both g_k and $\hat{g}_{k+1}$ for each k. There are four cases which apply depending on the nature of the game matrix with payoff $\hat{g}_{k+1}(i_k^e, i_k^p)$; $i_k^e = 0,1$; $i_k^p = 0, 1$. Case 1 applies if either diagonal dominates the matrix, and then there is a mixed strategy with optimal observation probabilities by solving $\partial \hat{g}_k / \partial \mu_k^e = 0$, and $\partial \hat{g}_k / \partial \mu_k^p = 0$, which will automatically satisfy the game theoretic saddle point condition. The result is

$$\hat{\mu}_k^e (I_{k-1}^e, I_{k-1}^p) = [\hat{g}_{k+1}(1,0) - \hat{g}_{k+1}(0,0)] / \sum_{i_k^e, i_k^p} (-1)^{i_k^e + i_k^p + 1} \hat{g}_{k+1}(i_k^e, i_k^p)$$

$$\hat{\mu}_k^p (I_{k-1}^e, I_{k-1}^p) = [\hat{g}_{k+1}(0,1) - \hat{g}_{k+1}(0.0)] / \sum_{i_k^e, i_k^p} (-1)^{i_k^e + i_k^p + 1} \hat{g}_{k+1}(i_k^e + i_k^p) \tag{10}$$

Case 2 occurs if $\min_{i_k^p} \max_{i_k^e} \hat{g}_{k+1}(i_k^e, i_k^p) = \max_{i_k^e} \min_{i_k^p} \hat{g}_{k+1}(i_k^e, i_k^p)$ i.e. no mixed strategy exists, and then $\hat{\mu}_k^e = i_k^e$, $\hat{\mu}_k^p = i_k^p$. There are two borderline cases. In case 3 one or more of the players has a nonunique solution in either or both of the min-max and max-min games, but the intersection of the solutions for these two games is unique, in which case $\hat{\mu}_k^e$ and $\hat{\mu}_k^p$ are just the i_k^e and i_k^p for this unique solution. In case 4 there is a nonunique solution satisfying both the min-max and max-min games on the $\hat{g}_k$ matrix, and in this case one or both of the players can pick any arbitrary observation probability $\hat{\mu}_k^e$ or $\hat{\mu}_k^p$ without influencing the total cost.

The above rules assign values to $\hat{\mu}_k^e$ and $\hat{\mu}_k^p$ which can be substituted into $g_k(\mu_k^e, \mu_k^p)$ to obtain $\hat{g}_k$ from $\hat{g}_{k+1}$. Recursive computation for k = N-1, N-2, . . ., 0 establishes the optimal observation probabilities $\hat{\mu}_k^e$ and $\hat{\mu}_k^p$ for all stages k, and for all possible past observation decision histories I_{N-1}^e, I_{N-1}^p, which is the desired result.

Probabilistic strategies for Class II are treated by the same techniques except that an augmented state equation for each player must be used, which are given in [2].

SEQUENTIAL DIFFERENTIAL GAMES WITH IS 5 AND IS 6

In the previous section probabilistic strategies were needed to obtain satisfaction of a saddle point condition for simultaneous decision problems. In Class III a totally different type of randomized strategy is introduced and will be useful for simultaneous problems as well. Consider the sequential IS 5 game in which at each stage k in the a priori choice of observation indicate the pursuer learns his opponents i_k^e before he chooses i_k^p, but the pursuer does not divulge his observation indicators to the evader. The pursuer can take advantage of this situation to further optimize his expected cost averaged over many games by allowing his observation choices to be random with an optimized probability. We will use the new symbol π_k^p for the probability that $i_k^p = 1$, in order to distinguish it from observation probabilities used previously for purposes of satisfying a saddle point condition in a simultaneous game. Also, let $\Pi_{N-1}^p = \{\pi_k^p, k = 0, 1, . . ., N-1\}$.

As before we will assume that each player's control law is an affine function of the data available to him in the form $u_k^p = a_k^p + \sum_j \Lambda^p(k,j) i_j^p z_j^p$. It will be convenient to define matrices for each player containing all controller gains used during a game. Let $\mathcal{L}^e$ be the lower partitioned triangular matrix whose partitions are the gain matrices $\Lambda^e(k,j)$; k = 0, 1, . . ., N-1; j = 0, 1, . . ., k, and let $\mathcal{A}^e$ be the column matrix whose kth partition is a_k^e. Analogous matrices are defined for the pursuer.

The quantities to be chosen a priori by the players are I_{N-1}^e, Π_{N-1}^e, and the matrices $\mathcal{L}^e, \mathcal{A}^e, \mathcal{L}^p, \mathcal{A}^p$. We assume that both players can make the computations that his opponent uses, so that the optimal values for each of these quantities is known to both players. Furthermore, since Π_{N-1}^p is chosen a priori, the pursuer can also choose his entire observation sequence I_{N-1}^p a priori for each game played. Therefore, the pursuer can use I_{N-1}^p

information as well as the above in his choice of control gains.

Consider first the pursuer's point of view. With the above assumptions, he can compute $\mathcal{B}^p$ and $\mathcal{A}^p$ as if he knows I^e_{N-1} , I^p_{N-1} , $\mathcal{B}^e$, $\mathcal{A}^e$. Equations in reference [2] give the needed equations. Note that there is no dependence on Π^p_{N-1} in this form. This dependence occurs through $\mathcal{B}^p$ and $\mathcal{A}^p$ dependence.

Now consider the evader's point of view. Since his control gains cannot depend on I^p_{N-1} , he must base his control choice on maximizing the expected cost averaged over the possible I^p_{N-1} , i.e. maximize

$$J^e(I^e_{N-1}, \Pi^p_{N-1}) = \sum_{\forall I^p_{N-1}} \prod_{j=0}^{N-1} \omega^p_j(i^p_j) J(I^e_{N-1}, I^p_{N-1}) \tag{11}$$

$$\omega^p_j(i^p_j) = \begin{cases} \pi^p_j & \text{if } i^p_j = 1 \\ 1-\pi^p_j & \text{if } i^p_j = 0 \end{cases}$$

over U^e_{N-1} (i.e. $\mathcal{B}^e$, $\mathcal{A}^e$) after substituting the affine forms for his opponent's control into J. This gives equations for the matrices $\mathcal{B}^e$, $\mathcal{A}^e$ as functions of the optimal values of $\mathcal{B}^p$, $\mathcal{A}^p$, I^p_{N-1} , and Π^p_{N-1} . Here we require that the control law for the evader can depend only on the a priori choice of Π^p_{N-1} , in order to prevent any attempt to use the data on-line to estimate conditional probabilities for the elements of Π^p_{N-1} . The four sets of equations obtained above define the control gains as functions of the form $\mathcal{B}^e(I^e_{N-1}, \Pi^p_{N-1})$, $\mathcal{A}^e(I^e_{N-1}, \Pi^p_{N-1})$, $\mathcal{B}^p(I^e_{N-1}, I^p_{N-1})$, $\mathcal{A}^p(I^e_{N-1}, I^p_{N-1})$ where the arguments must be optimal. The I^p_{N-1} would be chosen based on the optimal probabilities Π^p_{N-1} using a random number generator. It remains to choose the optimal values of I^e_{N-1} and Π^p_{N-1} . To do this, consider stage k and let $\hat{g}^p_{k+1}$ be the total a priori expected cost of the ensemble average of games, when any arbitrary I^e_k, I^p_k are used through step k, and values of $\hat{I}^e_{k+1,N-1} \triangleq \hat{I}^e_{N-1} - \hat{I}^e_k$ and $\hat{\Pi}^p_{k+1,N-1} \triangleq \hat{\Pi}^p_{N-1} - \hat{\Pi}^p_{k+1}$ are used for steps k+1 to N-1 which are optimal for this situation with carrot indicating optimal. Then for this situation, is chosen to minimize

$$\begin{aligned} g^p_k(i^e_k, \pi^p_k, I^{e/p}_{k-1}, \hat{I}^e_{k+1,N-1}, \Pi^p_{k-1}, \hat{\Pi}^p_{k+1,N-1}) = \\ (1-\pi^p_k)\, \hat{g}_{k+1}(i^e_k, i^p_k = 1, I^{e/p}_{k-1}, \hat{I}^e_{k+1,N-1}, \Pi^p_k, \hat{\Pi}^p_{k+1,N-1}) \\ + \pi^p_k\, \hat{g}_{k+1}(i^e_k, i^p_k = 0, I^{e/p}_{k-1}, \hat{I}^e_{k+1,N-1}, \Pi^p_k, \hat{\Pi}^p_{k+1,N-1}) \end{aligned}$$

After substituting the resulting $\Pi^p_k = \hat{\Pi}^p_k$ into the above, denote the result as $\hat{g}^e_k(i^e_k, I^{e/p}_{k-1}, \Pi^p_{k-1}, \hat{I}^e_{k+1,N-1}, \hat{\Pi}^p_{k,N-1})$. The evader does not know I^p_{k-1} but we can assume he knows Π^p_{k-1} since he can make the same calculations as the pursuer to determine $\hat{\Pi}^p_{N-1}$. Hence, he picks i^p_k to be 1 or 0, whichever maximizes

$$\sum_{\forall I^p_{k-1}} \prod \omega^p_j(i^p_j)\, \hat{g}^e_k(i^e_k, I^{e/p}_{k-1}, \Pi^p_{k-1}, \hat{I}^e_{k+1,N-1}, \hat{\Pi}^p_{k+1,N-1}) \tag{12}$$

The optimal i^e_k substituted into $\hat{g}^e_k$ gives $\hat{g}^p_k$. Then, to start this backwards recursive computation, one uses the fact that $\hat{g}^e_N(i^e_{N-1}, i^p_{N-1}, I^{e/p}_{N-2}, \Pi^p_{N-1}, -, -) = J(I^e_{N-1}, I^p_{N-1}, \Pi^p_{N-1})$. The right hand side is the total cost for a given pair of observation time histories, and a dependence on Π^p_{N-1} has been indicated this time to reflect the dependence of the control gains on this quantity.

As this recursive computation progresses, one obtains $\hat{\pi}^p_k$ and $\hat{i}^e_k$ as functions of Π^p_{k-1} for each k. At stage zero one obtains the optimal $\hat{\pi}^p_0$ and $\hat{i}^e_0$

associated with the desired problem. Progressing forward through these equations substituting the optimal values for Π_{K-1}^{p} gives the desired $\hat{\Pi}_{N-1}^{p}$ and I_{N-1}^{e}.

It is important to note the dependence on π_K^p in the $\hat{g}_{K+1}^{p}$ on the right hand side of (12), which appears as a result of the dependence of $\mathcal{S}^e$ and a^e on the optimal Π_{N-1}^{p}. It is this dependence in $\hat{g}_{K+1}^{p}$ which creates the fundamental difference between this type of probabilistic observation decision and that given earlier for purposes of satisfying a saddle point condition. This dependence also complicates the minimization of g_K^p to find π_K^p.

The compulations for IS 6 can be made similarly.

SIMULTANEOUS DIFFERENTIAL GAMES FOR IS 5 AND 6

When the observation decisions are to be made simultaneously at each stage k (and then the evader's result announced to the pursuer for IS 5, and vice-versa for IS 6) one wishes to satisfy a saddle point condition. As a result we will introduce a randomized observation strategy for the evader as well as the pursuer, using μ_K^e to represent the probability that is one, and letting $M_K^e \triangleq \{\mu_j^e, j = 1, 2, \ldots, k\}$, as before.

The quantities to be chosen a priori are M_{N-1}^e, Π_{N-1}^p, $\mathcal{S}^e$, a^e, $\mathcal{S}^p$, a^p. As before we can assume that each player can perform the computations of his opponent, so that both players know each of these quantities.

We consider IS 5. As before, the observation indicators are to be chosen off-line. Once the results have been obtained by both players at stage k, the evader announces his decision, and the players proceed to the next stage.

Consider the computation of the control gains, and first take the point of view of the pursuer. His control gains are to be determined a priori, and according to the above discussion he can wait until he knows the complete choice of I_{N-1}^{e} (as well as I_{N-1}^{p}) before he chooses these gains. The control gains for the evader can depend only on I_{N-1}^{e} and knowledge of other a priori calculated quantities. From these observations we see that as far as control gains are concerned each player faces the same problem as in the previous sequential section. Hence we have four sets of equations which define $\mathcal{S}^e(I_{N-1}^{e}, \Pi_{N-1}^{p})$, $a^e(I_{N-1}^{e}, \Pi_{N-1}^{p})$, $\mathcal{S}^p(I_{N-1}^{e}, I_{N-1}^{p})$, $a^p(I_{N-1}^{e}, I_{N-1}^{p})$. Again the control gains are restricted to use of a priori knowledge on Π_{N-1}^{p}.

To find the optimal M_{N-1}^e, Π_{N-1}^p define $\hat{h}_{K+1}(i_K^e, i_K^p, I_{K-1}^{e/p}, \hat{M}_{K+1,N-1}^{e}, \Pi_{K-1}^{p}, \hat{\Pi}_{K+1,N-1}^{p})$ to be the ensembel average expected cost of a game which uses I, I, and for steps o through k, and optimized probabilities $\hat{M}_{K+1,N-1}^{e}$, $\hat{\Pi}_{K+1,N-1}^{p}$ apply to latter stages. At stage k = N, $\hat{h}_N$ can be calculated by substituting the control gains for sequence pair I_{N-1}^{e}, I_{N-1}^{p} into $J(I_{N-1}^{e}, I_{N-1}^{p}, \Pi_{N-1}^{p})$. Recursive calculation backwards gives at step k

$$h_K(\mu_K^e, \pi_K^p, I_{K-1}^{e/p}, \hat{M}_{K+1,N-1}, \Pi_{K-1}^{p}, \hat{\Pi}_{K+1,N-1}^{p}) =$$
$$(1-\mu_K^e)(1-\pi_K^p)\,\hat{h}_{K+1}(0,0) + \mu_K^e(1-\pi_K^p)\,\hat{h}_{K+1}(1,0)$$
$$+ (1-\mu_K^e)\,\pi_K^p\,\hat{h}_{K+1}(0,1) + \mu_K^e\,\mu_K^p\,\hat{h}_{K+1}(1,1)$$

which is to be simultaneously maximized with respect to μ_K^e and minimized with regard to π_K^p. In the case of a mixed strategy one sets the following derivatives to zero, $\partial h_K/\partial \mu_K^e = 0$ and $\partial h_K/\partial \pi_K^p$. The first of these is a linear equation in μ_K^e alone which is easily solved. The second expression, being associated with the new type of probabilistic observations, gives a (generally) nonlinear equation involving both μ_K^e and π_K^p which is to be solved for μ_K^e using the already determined $\hat{\pi}_K^p$. Once k = 0 has been reached in the recursive computation, the problem of interest has been solved. The computations for IS 6 are analogous.

INFORMATION STRUCTURE 7

In the structure for IS 7 one notes that there is no difference between a priori sequential and a priori simultaneous observation decisions. The probabilistic strategies for each player at stage k, π_k^e and π_k^p , can depend on each player's knowledge of his opponent's complete set of observation probabilities, and on his own previous observation choices I_{k-1}^e and I_{k-1}^p respectively. Techniques similar to those of previous sections can apply to this problem.

REFERENCES

1. G. J. Olsder, "On Observation Costs and Information Structures in Stochastic Differential Games," Differential Games and Applications, Proceedings of a Workshop, Enschede, Holland, Springer-Verlag, 1977, pp. 172-185.

2. Z. T. Zhu, R. W. Longman and C. A. Cooper, "Optimal Observation and Control Strategies for Stochastic Differential Games with Costly Measurements," Proceedings of the 16th Annual Conference on Information Sciences and Systems, Princeton University, Princeton, New Jersey, March 1982, pp. 182-187.

3. Z. T. Zhu, R. W. Longman and C. A. Cooper, "Probabilistic Observation Strategies in Stochastic Differential Games with Costly Measurements," Proceedings of the 1982 American Control Conference, Arlington, Virginia, June 1982, pp.696-698.

TABLE 1. Information Structures

Class	Subclass	Pursuer Obtains		Evader Obtains	
		E's Observation Sequence	E's Measurement Sequence	P's Observ. Seq.	E's Meas. Seq.
I	IS 1	X	X	X	X
II	IS 2	X	X	X	
	IS 3	X		X	X
	IS 4	X		X	
III	IS 5	X			
	IS 6			X	
	IS 7				

ON THE LINEAR QUADRATIC CLOSED LOOP, NO MEMORY NASH GAME†

G.P. PAPAVASSILOPOULOS‡ AND G.J. OLSDER
Department of Applied Mathematics
Twente University of Technology
P.O. Box 217, 7500 AE Enschede
The Netherlands

ABSTRACT

We exhibit a class of LQ-Nash Games with the property that the solution exists for any finite time interval but for the infinite time case there exist none or a unique or many solutions depending on the choice of the parameters. Also, the limit of the finite time solution does not have to yield the infinite time case solution. A geometric formulation of the coupled algebraic Riccati equations is given. This formulation seems to be an interesting starting point for a thorough study of these equations.

INTRODUCTION

In this paper we consider two issues related to the Linear Quadratic Nash Game [1]. In the first we consider a class of problems which have the following properties. The finite time version admits a unique silution for any arbitrary but finite time interval whereas the infinite time version might have many, one or no solution. Moreover, even if the infinite time version has a solution, it does not have to be the limit of the finite time solution. A similar observation was demonstrated in [2] for a zero-sum linear quadratic game. In the second we provide a geometric formulation of the two Algebraic Riccati equations whose study is a central issue in solving Linear Quadratic Nash Games. This formulation is of interest and in our opinion it deserves further attention.

A LINEAR QUADRATIC CLOSED LOOP NO-MEMORY NASH GAME

Consider a dynamic system whose state x evolves according to

$$\dot{x} = Ax + Bu_1 + Bu_2, \quad x(0) = x_0, \quad t \quad [0,T] \tag{1}$$

and two cost functions

$$J_i(u_1,u_2) = x'(T)Q_{iT}x(T) + \int_0^T (x'Q_i x + u_i'u_i - u_j'u_j)dt, \tag{2}$$

$$i \neq j, \quad i,j = 1,2.$$

The matrices involved are constant, real, $Q_i \geqslant 0$, $Q_{iT} \geqslant 0$ and (A,B) is controllable. The admissible u_1, u_2 are linear in the current state.

Case i, $T < +\infty$:

It is known that a pair (u_1^*, u_2^*) is a Nash equilibrium if and only if

$$U_1^* = -B' K_i x, \qquad i = 1,2 \tag{3}$$

where

$$-\dot{K}_i = (A-BB'K_j)'K_i + K_i(A-BB'K_j) + Q_i - K_jBB'K_j - K_iBB'K_i \tag{4}$$

$$K_i(T) = Q_{iT}, \; i \neq j, \; i,j = 1,2 \qquad t \in [0,T]$$

---

† This work was supported in part by the U.S. Airforce Office of Scientific Research under Grants AFSOR-80-0171 and -82-0174.

‡ On leave of absence from: Department of Electrical Engineering-Systems, University of Southern California, Los Angeles, California 90089-0781, USA.

Adding and subtracting gives the equivalent system:

$$- \frac{d}{dt}(K_1+K_2) = A'(K_1+K_2) + (K_1+K_2)A + Q_1\ Q_2 + \\ -(K_1+K_2)2BB'(K_1+K_2) \tag{5}$$

$$K_1(T) + K_2(T) = Q_{1T} + Q_{2T}, \quad t \in [0,T]$$

$$- \frac{d}{dt}(K_1-K_2) = A'(K_1-K_2)+(K_1-K_2)A + Q_1 - Q_2 \tag{6}$$

$$K_1(T) - K_2(T) = Q_{1T} - Q_{2T}, \quad t \in [0,T]$$

Since (5) is an ARE for (K_1+K_2) and (6) is linear we conclude: that the finite time Nash Game (1)-(3) admits a unique linear solution for any arbitrarily large, but finite, time interval [0,T]. We can choose A, Q_i, Q_{iT} as to have that K_1, K_2 converge or become unbounded or oscillate.

Case ii: $T = +\infty$:

If a Nash equilibrium exists it will be of the form (3) where

$$0 = A'(K_1+K_2) + (K_1+K_2)A + Q_1 + Q_2 + \\ -(K_1+K_2)2BB'(K_1+K_2) \tag{7}$$

$$0 = A'(K_1-K_2) + (K_1-K_2)A + Q_1 - Q_2 \tag{8}$$

Since $J_1^* + J_2^* = \int_0^{+\infty} x'(Q_1+Q_2)x = x_0'(K_1+K_2)x_0$, and $Q_1 + Q_2 \geqslant 0$, $K_1 + K_2$ has to be positive semidefinite. Thus, assuming (A,B) controllable, and $(A,(Q_1+Q_2)^{\frac{1}{2}})$ observable, we conclude that $K_1 + K_2$ will be the unique positive definite stabilizing solution of (7). (8) might have one or many or no solutions. The basic conclusion is: the System (7)-(8) might have one or many or no solutions. Even if it has a solution, this solution does not need to be the limit of the solutions of (5)-(6).

That the limit of the finite time solution, as T → +∞, does not have to be the infinite time solution, has also been observed in [2] for the case of zero-sum games.

A GEOMETRIC FORMULATION OF THE COUPLED ALGEBRAIC RICCATI EQUATIONS

It is known that the problem of finding all the symmetric solutions of the algebraic Riccati equation

$$0 = A'K + KA + Q - KBB'K \tag{9}$$

can be reformulated equivalently as follows.

Find the n-dimensional subspaces $\mathcal{V}$ in R^{2n} for which it holds

$$A_s\mathcal{V} \subseteq \mathcal{V} \qquad \left(A_s = \begin{bmatrix} A & -BB' \\ -Q & -A' \end{bmatrix}\right) \tag{10}$$

If the columns of the 2nxn matrix

$$V = \begin{array}{cc} V_1 & n \\ V_2 & n \end{array}$$

constitute a basis for $\mathcal{V}$ and V_1^{-1} exists and $V_2V_1^{-1}$ is symmetric one has a solution $K = V_2V_1^{-1}$ of (9).

Thus, the whole issue is the appropriate choice of n eigenvectors and generalized eigenvectors of A_S which will form V.

Let us now generalize the previous formulation to the Linear Quadratic, Closed-Loop, No-Memory Nash Game. Let the system equation and the two costs be given by

$$\dot{x} = Ax + B_1u_1 + B_2u_2, \quad J_i = \int_0^\infty (x'Q_ix + u_i'u_i + u_j'R_{ij}u_j)dt, \quad i \neq j, \ i,j = 1,2.$$

If a pair (u_1,u_2) is a Nash equilibrium where each u_i at time t is a linear function of $x(t)$, it has to be given by (3), where the matrices K_1, K_2 are symmetric and satisfy the system:

$$0 = K_i(A-B_jB_j'K_j) + (A-B_jB_j'K_j)'K_i + Q_i + K_jB_jR_{ij}B_j'K_j \quad -K_iB_iB_i'K_i \tag{11}$$

Let

$$A_s^N = \begin{bmatrix} A & -B_1B_1' & -B_2B_2' \\ 0 & B_1R_{21}B_1' & -B_1B_1' \\ 0 & -B_2B_2' & -B_2R_{12}B_2' \\ -Q_1 & -A' & 0 \\ -Q_2 & 0 & -A' \end{bmatrix}, \quad L_1 = \begin{bmatrix} 0 & 0 & 0 \\ I & 0 & 0 \\ 0 & 0 & 0 \\ 0 & I & 0 \\ 0 & 0 & I \end{bmatrix}, \quad L_2 = \begin{bmatrix} 0 & 0 & 0 \\ I & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & I & 0 \end{bmatrix}, \quad L_3 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ I & 0 & 0 \\ 0 & 0 & I \\ 0 & 0 & 0 \end{bmatrix}$$

We can show, that solving (10) is equivalent to the following problem: find the n-dimensional subspaces M of R^{3n} for which it holds

$$A_s^N M \subseteq L_1M + L_2M + L_3M \tag{12}$$

If $M = [M_0' \ M_1' \ M_2'](3nxn)$ is a basis for M, where M_0^{-1} exists and $M_1M_0^{-1}$, $M_2{}^{M}{}_0^{-1}$ are symmetric, then $K_1 = M_0M_0^{-1}$, $K_2 = M_2M_0^{-1}$ solve (10).

We thus need to solve (12) which is a generalization of (10). Although we do not know at present how to do that, it seems that this formulation of the problem is quite appealing and deserves further attention. Notice also that the role of the system matrix A_S of the control problem, is assumed in the game case by A_S^N in conjunction with L_1, L_2, L_3.

REFERENCES

1. T. Basar and G.J. Olsder, "Dynamic Non-Cooperative Game Theory," Academic Press, 1982.

2. E. Mageirou, "Values and Strategies for Infinite Time Linear Quadratic Games," IEEE Transactions on Automatic Control, Vol. AC-21, pp. 547-550, 1976.

OPTIMAL DECENTRALIZED DISCRETE TIME SYSTEMS

KOUROSH RAHNAMAI and MAHMOUD SAWAN
Electrical Engineering Department
Wichita State University
Wichita, KS 67208
(316) 689-3415

ABSTRACT

Problem of synthesizing discrete decentralized output regulators for large scale systems is considered. The problem is solved by posing it in terms of the optimization of a discrete linear regulator with a quadratic performance index.

INTRODUCTION

Because of increasing interest in discrete-time systems for its application in digital control and also promising decentralized method for controlling large scale systems a discrete decentralized system has been considered and an attempt is made to optimize the system.

DERIVATION OF MAIN RESULT

Consider the system

$$X(K+1) = AX(K) + \sum_{i=1}^{L} B_i U_i(K) \tag{1}$$

with

$$Y_i(K) = C_i X(K) \qquad i=1,2,\dots,L. \tag{2}$$

Where $X(K)\varepsilon\mathbb{R}^{n}$, $U_i(K)\varepsilon\mathbb{R}^{m_i}$, $Y_i(K)\varepsilon\mathbb{R}^{r_i}$ and L is the number of subsystems. It is desired to design decentralized output feedback which minimizes the quadratic performance criterion

$$J = \frac{1}{2}\sum_{K=0}^{\infty} \{X'(K)QX(K) + \sum_{i=1}^{L} U_i'(K)R_{ii}U_i(K)\} \tag{3}$$

The output feedback can be written as

$$U_i(K) = -F_i Y_i(K). \tag{4}$$

It is required to find F_i such that J is minimized. The cost functional may be written as

$$J = \frac{1}{2} X_0'(K) M X_0'(K) \tag{5}$$

Where the cost matrix M satisfies the Lyapunov equation

$$M = A'MA + Q + \sum_{i=1}^{L} W_i(M,F_i) \tag{6}$$

Where

$$W_i(M,F_i) = C_i'F_i'R_{ii}F_iC_i - C_i'F_i'B_i'MA - A'MB_iF_iC_i + C_i'F_i'B_i'MB_iF_iC_i \tag{7}$$

In order to find a matrix M which minimizes the cost criterion, $W_i(M,F_i)$ should be minimized with respect to F_i. This can be done by setting the differential of the trace of $W_i(M,F_i)$ to zero, therefore

$$R_{ii}F_iC_iC_i' + B_i'MB_iF_iC_iC_i' - B_i'MAC_i' = 0 . \tag{8}$$

Solving equation (8) for F_i results in

$$F_i = (R_{ii} + B_i'MB_i)^{-1} B_iMAC_i'(C_iC_i')^{-1} \tag{9}$$

Control for each subsystem is given by equation (9). Therefore in order to find the optimum input for each subsystem equations (9), (7), and (6) should be solved iteratively.

EXAMPLE

Consider a second order system with

$$A = \begin{bmatrix} 0 & 1 \\ -0.02 & 0.3 \end{bmatrix}, \quad B_1 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \quad B_2 = \begin{bmatrix} 0 \\ 2 \end{bmatrix}, \quad M_0 = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$$

$$C_1 = \begin{bmatrix} 0 & 1 \end{bmatrix}, \quad C_2 = \begin{bmatrix} 1 & 0 \end{bmatrix}, \quad Q = I \quad \text{and } R_{11} = R_{22} = 1$$

Using these values, equations (6), (7) and (9) have been solved iteratively and the values of feedback gains are found to be

$$F_1 = 0.204338 \quad \text{and} \quad F_2 = -0.008919$$

OLIGARCHICAL CONTROL OF DISTRIBUTED PROCESSING SYSTEMS

MOSHE KRIEGER
Department of Electrical and Computer Engineering
Syracuse University, Syracuse, NY
on-leave from Department of Electrical Engineering
University of Ottawa
Ottawa, Ontario, Canada

NICOLA SANTORO
Distributed Computing Group
School of Computer Science
Carleton University
Ottawa, Ontario, Canada

ABSTRACT

When designing a distributed processing system, a control mechanism is needed to enforce and maintain the proper coordination, functioning and communication between the processing elements. In this paper, we introduce the concept of oligarchical control which integrates both centralized and decentralized control features. We show how this concept can be used to design control mechanisms which are highly reliable, offer fast response time for control operations, and require simple coordination elements.

1. INTRODUCTION

One of the most crucial problems faced by the designers of distributed processing systems is to provide the control mechanism to enforce and maintain the "proper" coordination, functioning and communication between the processing elements (8,9,11). The choice is usually between two extreme cases: fully centralized and fully decentralized mechanisms, each offering attractive but distinct and often conflicting features; depending on the choice, the system will be called fully centralized or fully decentralized.(6)

In fully decentralized systems, a coordinator is associated with each entity, and the coordinators are able to communicate with all the other coordinators in the system. The decision function is obtained by performing distributed computations which usually require the participation of all coordinators and involve a large number of information (data and/or control) transfers. The main disadvantage in these systems is that considerable amount of (processing and communication) time is required to perform the decision function; software overhead is incurred for communication protocols, and there are also delays in processing due to waiting time in queues at both the sending and receiving end of each communication path. The extra processing and waiting time due to interprocessor communication grows rapidly with the size of the system, and any system of non-trivial size will experience a degradation in the global throughput. The main advantages of these systems are that the coordinators can be very simple, and the overall system operability (i.e., the system ability to function) can be ensured, to a large degree, in spite of coordinator failures.

In fully centralized systems, there is only one coordinator to which each entity must be able to communicate. This fact allows fast response time for control operations; however, the overall system operability relies on the proper functioning of the only coordinator, making the system highly vulnerable. Furthermore, the complexity (e.g., processing capacity, memory size, number of communication ports, etc.) of the coordinator increases very rapidly with the number of elements in the system. To reduce the complexity of the coordinator in very large systems, the idea of hierarchical control has been introduced (1,10): the control mechanism is organized as a tree of coordinators with different degrees of complexity, the entities being the leaves of the tree.

In this paper we refine the idea of hierarchical control integrating features from both decentralized and centralized mechanisms; and present the idea of oligarchical control which allows the design of highly reliable systems with fast response time for control operations, and which are easy to implement.

In the following sections, we characterize the basic components of a distributed processing system, define oligarchical systems, and analyze their complexity.

2. PROCESSING AND COORDINATION

A distributed processing system can be viewed as composed of processing elements, called entities, each entity provided with its local control unit; and of system control elements, called coordinators. Entities perform processing activities, which include actual data manipulations and (possibly) entity control operations. Coordinators perform system control activities which include inter-entity coordination and involve exchange of control information (not data).

The distinction between entities and coordinators is a functional one; in practice, a physical system component can be provided with both processing and coordination capabilities. Using this distinction we can express a control mechanism as a decision function which maps requests originated by a system elements (both coordinators and entities) into a sequence of primitive commands which will activate processing activities by the appropriate entities.

The scope and operation of a control mechanism obviously depends on the nature of the system being designed. For instance, in distributed databases (2), the goal of the control mechanism is to enforce and maintain the mutual and internal consistency of the "fragments" of the databases; upon reception of an update request, the concurrency control mechanism will send the appropriate command (to perform or reject the update) to the database managers (the entities) of the sites where the database fragments involved in the update reside. In parallel and multi-task computers (5), the primary objective is to coordinate the activities of the executors involved in the processing of the same process; upon reception of a request to perform a task (e.g., multiplication of two matrices), commands are sent to the executors (the entities) specifying the location of the source data and of the (partial) results. In industrial automation (3), the main concern is to provide control of each work station and coordination among cooperating work stations; upon reception of a request by a work station (e.g., a robot arm), commands are sent to the other work stations (e.g., vision systems) whose cooperation is needed to complete the original task.

In the next section, we present a general model for control mechanisms this model can be used in systems very different in nature (as ones described above) provided that processing and coordination activities are separated.

3. OLIGARCHICAL SYSTEMS

3.1 TOPOLOGY

In the oligarchical organization, the set L_o of entities constitutes the base of the system. The control mechanism is organized by levels: there are $n \geq 1$ levels of "control" and two "modes" of operation for each level. Each level i, $1 \leq i \leq n$, is composed of a set L_i of coordinators; each coordinator $x \varepsilon L_i$ is directly connected to several coordinators at level i-1 (in case i=1, the coordinators are connected to system entities), and to at least one coordinator at level i+1 (except in case i=n).

Additional specifications of the topology of an oligarchical system are:

1. $|L_i| \leq |L_{i-1}|$; i.e., the number of coordinators is decreasing in the levels;
2. $|L_n| \geq 1$; i.e., the top level can have more than one coordinator, called managers;
3. Each coordinator $x \varepsilon L_i$ can communicate (either directly or indirectly) to all other coordinators in L_i;
4. Each level is "globally connected" to the next levels; i.e., each coordinator in L_i can communicate (either directly or via another coordinator in L_i) to any coordinator in L_{i+1}, and vice versa.

An example of topology of an oligarchical system is shown in Figure 1.

The level of a coordinator as well as the size of the subsystem below it of which it is aware determines its control capability. Formally the view $v(x) \leq L_o$ of a coordinator $x \varepsilon L_i$ is defined as the set of entities directly connected to x if i=1; otherwise, as the union of views of all coordinators $y \varepsilon L_{i-1}$ directly connected to x.

3.2 COMMUNICATION AND CONTROL

There are three kinds of (control) communication primitives: requests, control signals (or commands), and transfers. Requests are sent to the higher level, control signals are sent to the lower level, and transfers are sent within the same level. We shall denote by R_i and C_i the set of requests and commands which can be sent from level i; by R_o we denote the set of requests sent by the system entities. For a set of commands C_i, let $S(C_i)$ denote the set of all finite sequences of commands in the set.

Requests typically originate at the system entities level. Associated to each request $\rho \ \varepsilon \ R_i$ there is a scope $s(\rho) \leq L_o$ which specifies the degree of decisional capability needed to resolve. When a request is resolved at level i, appropriate commands are sent to the lower level as specified by the control function $f_i: R_{i-1} \rightarrow S(C_i)$. When a command is received at level i, the appropriate action is taken as specified by the processing function $g_i: C_{i+1} \rightarrow S(C)$, $i \geq 1$. When an entity $e \ \varepsilon L_o$ receives a command $c \ \varepsilon C_1$ it will perform the appropriate processing activities.

Each level can operate in two different modes: centralized and decentralized. In the centralized mode, when a requests $\rho \ \varepsilon R_{i-1}$ arrives at $x \varepsilon L_i$ if $s(\rho) \leq v(x)$ (i.e., x has the control capability of resolving the request) it does do; otherwise, it sends the appropriate request to level i+1 for resolution. In a decentralized (or, survival) mode, any request arriving from level i-1 can be resolved collectively by the coordinators in L_i in a distributed fashion.

In other words, a coordinator at level i is capable of resolving immediately certain requests; for the other requests, it can either defer the resolution to a higher level or resolve it in a distributed fashion with the contribution of the other coordinators in L_i. The decision on which mode of operation should be used is made according to the following rule: *coordinators at level i operate in centralized mode unless there is a disconnection between levels i and i+1*. Level disconnection can be defined to occur when there is a *total* disconnection (i.e., absence of any route between two levels) or just *functional* disconnection (eg., the average or maximum route length between the two levels exceeds a fixed "performance degree").

Let us stress that the rules according to which a request is resolved or deferred are the same for all coordinators independently of the level. This implies that all coordinators are functionally *equivalent* in that they execute the same algorithm only on different data; thus, they can all be physically identical.

4. COMPLEXITY OF OLIGARCHICAL SYSTEMS

4.1 GENERAL CONSIDERATIONS

In the previous sections, we have specified only the minimum requirements that the topology must meet; to each topology meeting these requirements corresponds an oligarchical system. Regardless of the actual topology, some general facts about oligarchical systems can be observed.

A first observation is that both fully centralized and fully decentralized systems can be seen as instances of oligarchical systems. A fully centralized system is an oligarchical system with only one level of coordinators containing only one element, which can resolve any request. A fully decentralized system is an oligarchical system with only one level of coordinators; each coordinator is directly linked to one and only one entity, it is connected (either directly or indirectly) to all other coordinators, and it can resolve individually only certain requests, while all other requests require a switch to distributed mode.

A second observation relates to the separation between processing and coordination activities which are traditionally integrated under the ubiquitous term "control". One of the reasons for the existing situation has been the lack of clear distinction between entity-control and system-control functions; while both relate to the control of entities, the former regulates the functioning of an entity as a unit, the latter manages the cooperation among entities. This lack of distinction, tollerable in small systems, becomes a major obstacle in large systems. The introduction at entity level of local controllers, which include all the necessary entity control primitives, allows us to solve separately the problems related to system control (i.e., coordination).

In an actual implementation this separation, which lies at the basis of the oligarchical concept, has the immediate effect of reducing the amount of data transfers (reducing, in turn, time delays) to a minimum. Take for example the concurrency control problem in distributed databases; in almost all existing solution mechanisms both coordination information (e.g., entity names, time-stamps, etc) *and* data (entity values) are transmitted to resolve an update request. In an oligarchical system to resolve a request, only the coordination information is transmitted; the actual data will be sent if and only if the request has been accepted, and only "horizontally" (i.e., within entity level). That is, the above separation implies a functional distinction between "status" and "value" of each request for a decision originated at an entity; with only the former being used in the decision process.

Finally, in an oligarchical system, coordination is achieved according to the inverse of "Peter's Principle"; i.e., each request reaches its level of competence (or resolubility).

4.2 RELIABILITY CONSIDERATIONS

Although the actual complexity of an oligarchical system depends on its topology, certain basic reliability properties are common, provided that there is more than one coordinator at level 1 (and, possibly, more than one level). In fact, under these assumptions, the system achieves a very high degree of reliability.

In case of failure of a coordinator $x \in L_i$, any coordinator $y \in L_{i-1}$ directly connected to x can still send its requests to another coordinator $z \in L_i$ either directly or via some $w \in L_{i-1}$; recall, the coordinators in a given level can communicate with each other. Similarly, in case of direct line failure between $y \in L_{i-1}$ and $x \in L_i$, y can still send its requests to level i.

If there is a disconnection between two levels, i and i+1, the coordinators in L cannot send their requests to level i+1. In this case, they switch to a distributed mode of operation which allow them to resolve any request. Thus the system will still be operational albeit at a slower pace.

Depending on the level and on the mode, fail-safe or fail-soft operation of the system can be achieved; the former at the lowest level of coordination, the latter at higher ones.

4.3 EFFICIENCY CONSIDERATIONS

To analyse the efficiency of an oligarchical system we need to fully specify its topology. In this section we introduce an abstract topology, which can be used as a basis for the design of a large class of oligarchical systems. For this class we will evaluate the complexity in terms of the "time delays" for processing control operations and of the "size" of the resulting systems; and compare these results with the fully decentralized alternative.

The abstract topology can be defined constructively as follows. Let s be the number of entities; r be a positive integer, $1 \leq r \leq s$, called reduction factor and n be the number of desired levels. To simplify the following discussion, a fully symmetrical structure is assumed with $s=r^n \cdot m$, for some positive integer $m \geq 1$.

(1) Partition the entities in groups of r elements and associate a coordinator to each group; these s/r coordinators form the first level L_1.
(2) Recursively, apply the same partitioning technique to L_{i-1} to construct L_i; end the process when L_n has been constructed.
(3) At each level, connect the elements so to provide full reachability within the level.

This topology is characterized by the three parameters s, r and n. At each level i, the number of coordinators is s/r^i; at the top level there are $m=s/r^n$ managers. We shall denote by OL(s,r,n) the systems associated with this abstract topology. The fully decentralized systems correspond to OL(s,1,1) and the fully centralized correspond to OL(s,s,1).

The complexity of any distributed system is characterized by the number of coordinators (the size), and the maximum time delay needed to resolve a request when the system is properly operating. In oligarchical systems, the maximum time delay in absence of level disconnection occurs when a request, originated at the entity level, is resolved at manager level in a distributed way. Let K(s,r,n) and T(s,r,n) denote the size and the maximum time delay of OL(s,r,n).

Property 1

$$K(s,r,n) = \begin{cases} 1 & \text{if } r=n=1 \\ s(r^n-1)/r^n(r-1) & \text{otherwise} \end{cases}$$

This follows from $K(s,r,n)=\sum_{i=1}^{n} (s/r^n)$.

Property 2

$$T(s,r,n) = 2c(n)+d(s/r^n)$$

where c(x) is the time delay for a message (request or command) to pass through x levels, and d(y) is the time delay needed by y elements to resolve a request in a distributed fashion. The actual value of T depends on the functions c and d. Both functions can be assumed to be linear in their arguments; i.e., $c(x)=ax$ and $d(y)=by$, for some constants a and b. This assumption is obviously reasonable for c, and it can be achieved in practice for d by using a ring interconnection within the manager level; in practice $b \geq a$ and, depending on the complexity of the distributed decision logic, b can be much larger than a.

Since $s=r^n \cdot m$, then $n \leq \log_r s$; that is, even in very large systems, the number of levels will be very small. Furthermore, both the time delay and the size can be drastically smaller than the corresponding values in a fully decentralized system. In fact, T and K are bounded as follows:

Property 3

(i) $T(s,r,n) \leq 2 \cdot a \cdot \log_r s+b\ m$

(ii) $K(s,r,n) \leq (s-1)/(r-1)$ for $r > 1$

To illustrate the advantages of oligarchical system, the values of T and K for various s, r, and n have been computed and are shown below, where k=b/a.

OL(s,1,1) fully decentralized

s,r,n	250,1,1	500,1,1	1000,1,1	2000,1,1
T	250k	500k	1000k	2000k
K	250	500	1000	2000

OL(s,2,n)

s,r,n	256,2,6	512,2,7	1024,2,8	2048,2,9
T	12+4k	14+4k	16+4k	18+4k
K	232	488	1000	1924

OL(2,3,n)

s,r,n	243,3,4	486,3,4	972,3,5	2187,3,6
T	8+3k	8+6k	10+4k	12+3k
K	120	240	484	1092

OL(s,4,n)

s,r,n	256,4,3	512,4,3	1024,4,4	2048,4,4
T	6+4k	6+8k	8+4k	8+8k
K	84	168	340	680

OL(s,5,n)

s,r,n	250,5,3	500,5,3	1000,5,3	2000,5,3
T	6+2k	6+4k	6+8k	6+16k
K	62	160	248	496

A graphical representation of the difference in complexity between fully decentralized and the other oligarchical systems is presented in Figures 2 and 3. These results are self-explanatory and indicate the relevance of the oligarchical approach in systems of non trivial size.

5. CONCLUDING REMARKS

In this paper we have introduced the concept of oligarchical control, a general scheme for controlling the activities of a (very) large number of processing elements. At the basis of the oligarchical concept lies the separation between processing and coordination activities; an oligarchical control mechanism can be constructed in most systems by properly defining the entity controllers. This concept of separation and its need has emerged from our experiences in designing a number of systems and control mechanisms of various complexity: general-purpose uniprocessor (14), centrally-controlled segmented bus (4), task-driven multimicrocomputer (7), implicit routing (12), and concurrency control for fully duplicated distributed databases (13).

There are several factors which must be taken into account when implementing an oligarchical control mechanism. For example, to increase the system performance the partitioning of the entities should be done according (to) their level of (expected) interaction; entities whose cooperation is often required (e.g., vision system and robot's arm) should be assigned to the same coordinator. Accordingly, the partitioning of coordinators should reflect the degree of interaction of the entities within its view. Also, the overall system reliability can be greatly enhanced by connecting each system element to more than one coordinator at the next level; this obviously will increase the complexity of the coordinators and the number of connecting lines.

Presently, we are investigating the use of the oligarchical concept in partially duplicated distributed databases, high-level-language distributed computer systems, computer-aided manufacturing, local area networks, and VLSI chips.

REFERENCES

(1) J.S. Albus, A.J. Barbera, R.N. Nagel: "Theory and Practice of Hierarchical Control", Proc. National Computer Conf. 1981.

(2) P.A. Bernstein, N. Goodman: "Concurrency Control in Distributed Database Systems", ACM Computing Surveys, Vol. 13, No. 3, 1981.

(3) M.R. Buchner, I. Lefkowitz: "Distributed Computer Control for Industrial Process Systems: Characteristics, Attributes and an Experimental Facility", Control System Magazine, 1982.

(4) E.T. Fathi, M. Krieger "Centrally Controlled Segmented Bus Architecture", 19-th Annual Allerton Conf. on Communication, Control and Computing, 1981.

(5) Z. Gaili, W.J. Paul: "An Efficient General Purpose Parallel Computer", Proc. 13-th ACM Symp. on Theory of Computing, 1981.

(6) E.D. Jensen: "Distributed Control in Distributed Systems - Architecture and Implementation", Springer-Verlag, Lecture Notes in Computer Science 105, 1981.

(7) M. Krieger, E.T. Fathi: "Design Aspect of a Simple Distributed Microprocessor System", - Int. Conf. Communications, Circuits and Systems, Calcutta, 1981.

(8) M. Krieger, N. Santoro: "Some Open Questions in (Complex) Computer System Design - A Cry For Help", Int. Conf. on Advances in Information Sciences & Technology, Calcutta, 1982.

(9) B.W. Lampson, M. Paul, H.J. Siepert (edrs): "Distributed Systems - Architecture and Implementation", Springer-Verlag, Lecture Notes in Computer Science 105, 1981.

(10) M.S. Mahmoud: "Multilevel System Control and Applications: A Survey", IEEE Trans. on Systems, Man and Cybernetics. Vol. SMC 7, No. 3, 1977.

(11) C.V. Ramamoorthy, S.T. Doug, S.L. Ganesh, C-H Jev, W-T Tsai: "Architectural Issues in Distributed Systems", Int. Conf. on Communications, Circuits and Systems, Calcutta, 1981.

(12) N. Santoro, R. Khatib: "Labelling and Addressing in Distributed Systems", Proc. 19-th Allerton Conf. on Comm., Control and Computing, 1981.

(13) N. Santoro, H. Ural: "Concurrency Control in Large Computer Networks", Proc. IEEE Int. Large Scale Systems Symp., 1982.

(14) S. Vaillancourt, M. Krieger: "Segmented Microprogramming", Princeton Conf. on Information Sciences & Systems, 1982.

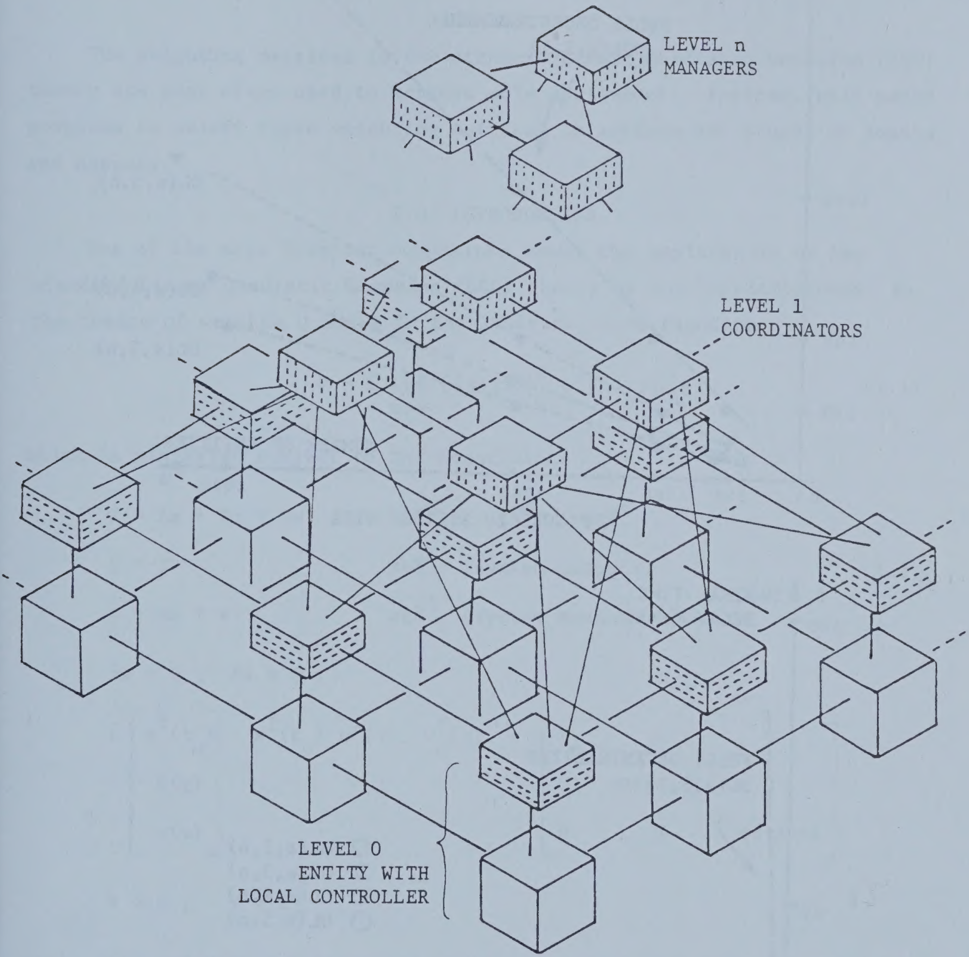

LEVEL n
MANAGERS
LEVEL 1
COORDINATORS
LEVEL 0
ENTITY WITH
LOCAL CONTROLLER

FIGURE 1 AN OLIGARCHICAL STRUCTURE

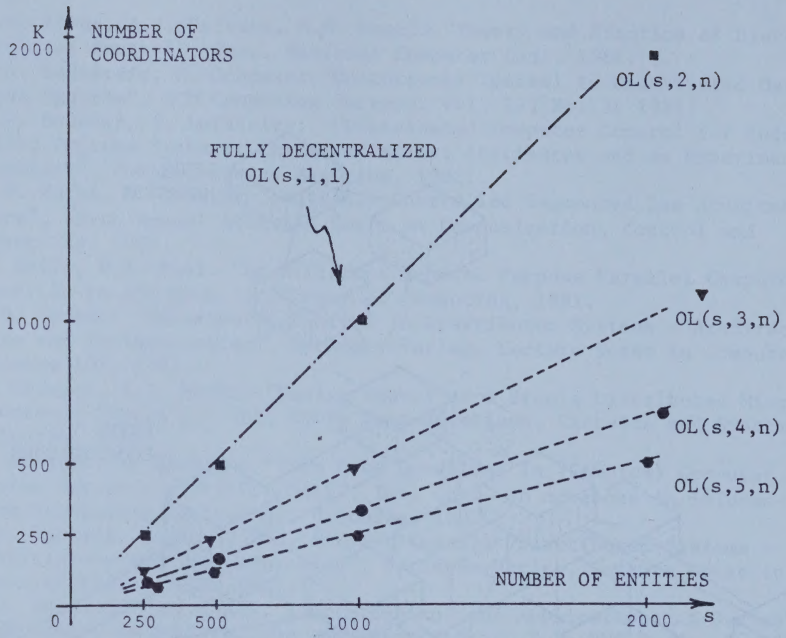


FIGURE 2 SYSTEM SIZE

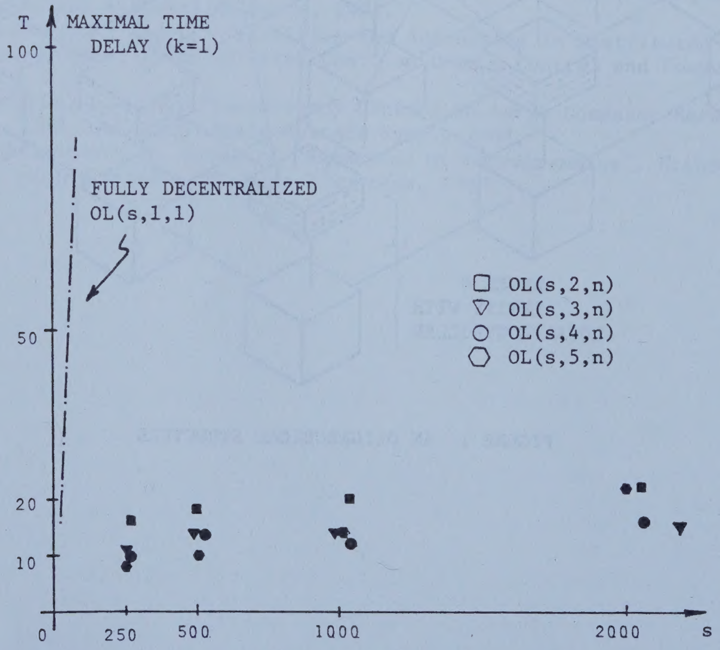


FIGURE 3 MAXIMAL DELAY TIME

ON SELECTION OF WEIGHTING MATRICES IN THE LQG PROBLEM

R.E. SKELTON AND M.L. DE LORENZO
Purdue University
School of Aeronautics and Astronautics
West Lafayette, Indiana 47907

ABSTRACT

The weighting matrices in the standard Linear Quadratic Gaussian (LQG) theory are most often used to achieve pole assignment. Instead, this paper proposes to select these weighting matrices to achieve RMS bounds on inputs and outputs.

1.0 INTRODUCTION

One of the most frequent complaints about the application of the standard Linear Quadratic Gaussian (LQG) theory is the "arbitrariness" in the choice of weights Q and R in the quadratic cost function

$$V = E \lim_{t\to\infty} \left(\|y(t)\|_Q^2 + \|u(t)\|_R^2 \right) \tag{1.1}$$

which is minimized subject to the constraints

$$\begin{aligned} \dot{x} &= Ax + Bu + Dw \;, && x \epsilon R^n \;, \quad u \epsilon R^m \;, \quad w \epsilon R^p \\ y &= Cx && y \epsilon R^k \quad \text{(system outputs)} \\ z &= Mx + v && z \epsilon R^{\ell} \quad \text{(system measurements)} \end{aligned} \tag{1.2}$$

$$Ew = 0 \;, \quad Ev = 0 \;,$$

$$E \begin{bmatrix} x^T(t_o) \\ w(t) \\ v(t) \end{bmatrix} \begin{matrix} x^T(t_o) & w^T(\tau), & v^T(\tau) \end{matrix} = \begin{bmatrix} X_o & 0 & 0 \\ 0 & W\delta(t-\tau) & 0 \\ 0 & 0 & V\delta(t-\tau) \end{bmatrix}$$

$$W > 0 \;, \quad V > 0$$

where w(t) and v(t) are zero-mean white noise processes with intensities W and V respectively, and E is the expectation operator and T denotes matrix transposition. In the literature, there have been two basic suggestions for the choice of Q, R.

The first suggestion, made by Bryson [1], relates Q and R to the input-output specifications σ_i^2, μ_i^2; where the *desired* values of the response of the closed-loop system are

$$y_i^2(t) \le \sigma_i^2 \quad \text{or} \quad \left(\frac{y_i}{\sigma_i}\right)^2 \le 1$$
$$u_i^2(t) \le \mu_i^2 \quad \text{or} \quad \left(\frac{u_i}{\mu_i}\right)^2 \le 1 \tag{1.3}$$

leading to Bryson's suggestion for the choices of Q and R

$$Q = \begin{bmatrix} 1/\sigma_1^2 & & \\ & \ddots & \\ & & 1/\sigma_k^2 \end{bmatrix}, \quad R = \begin{bmatrix} 1/\mu_1^2 & & \\ & \ddots & \\ & & 1/\mu_m^2 \end{bmatrix} \tag{1.4}$$

Hence with this choice of Q and R (1.1) becomes

$$V = \lim_{t\to\infty} \{ \sum_{i=1}^{k} Q_{ii}(Ey_i^2) + \sum_{j=1}^{m} R_{jj}(Eu_j^2) \} \tag{1.5}$$

It usually happens, however, that even though this choice of Q and R has some physical motivation the *actual* $y_i^2(t)$, $u_i^2(t)$ do not satisfy the *desired* bounds (1.3). Furthermore, there is no theory available which will show how to choose Q and R such that (1.3) is satisfied. Neither is there a method available which will guarantee satisfaction of (1.3) in the mean squared sense

$$Ey_i^2(t) \le \sigma_i^2 \ , \quad i = 1, \ldots, k$$
$$Eu_j^2(t) \le \mu_j^2 \ . \quad j = 1, \ldots, m \tag{1.6}$$

The second basic suggestion about the choice of Q, R in the literature has been to choose Q and R to achieve a desired pole assignment in the closed-loop system [2],[3]. In [4] a gradient technique adjusts the elements of Q, R to maximize the singular values of selected return difference matrices which are related to stability margins and disturbance rejection. There has been much more written about the pole assignment role for Q and R than about the time response assignment (1.3) or the variance assignment (1.6). This is indeed curious since LQG is a theory steeped in the *time-domain* and is *directly* concerned with the time responses y(t) and u(t). One might even argue that it is a bit obtuse to use LQG theory to do pole assignment since there are more direct methods to assign poles which do not use the artifice of LQG theory, [5].

This paper deals with the more natural use of LQG theory in the time-domain question relating to objective (1.6). The ideal goal is to find a linear dynamical feedback controller which satisfies the variance constraints (1.6). We shall alter this problem statement to make it more tractable by using the steady approximation of the constraints (1.6)

$$\begin{aligned} \lim_{t\to\infty} \mathcal{E}y_i^2(t) &\le \sigma_i^2 \\ \lim_{t\to\infty} \mathcal{E}u_i^2(t) &\le \mu_i^2 \end{aligned} \tag{1.7}$$

Then, a penalty function approach (i.e. LQG theory) is employed to accommodate the constraints (1.7), in the manner (1.5) by proper choice of Q_{ii}, and R_{ii}. This formulation automatically satisfies the constraint for a linear dynamical feedback controller, since the optimal LQG controller for (1.5) employs the standard Kalman filter and state feedback control gains. This will be called the

Constrainted Variance LQG Problem (LQG_{cv}):

Find Q_{ii} *and* R_{ii} *in* (1.5) *so that* (1.7) *is satisfied, subject to* (1.2).

The LQG_{cv} problem does not have a solution for every set of requirements σ_i^2, $(i=1, \ldots, k)$, μ_j^2, $(j=1, \ldots, m)$. To see that this is true, consider a single input-single output system. Hence, $y\varepsilon R^k (k=1)$, $u\varepsilon R^m (m=1)$. The LQG theory promises an inverse relationship between output regulation and control effort as the weighting R on control effort varies from 0 to infinity, as depicted in Fig. 1.

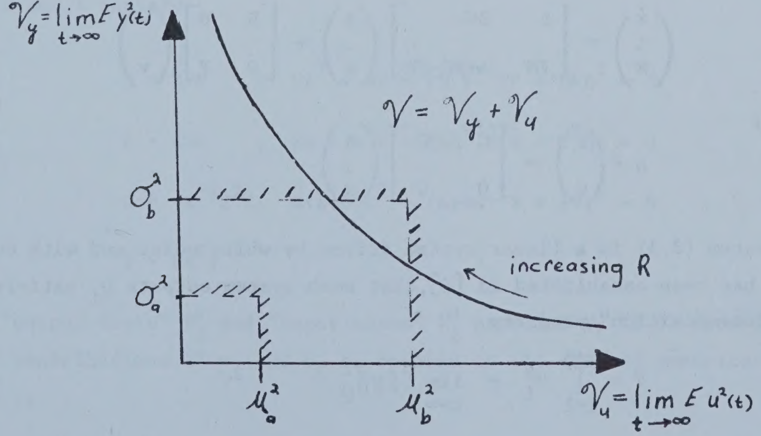


Fig. 1 Results of LQG theory for SISO Systems

Clearly no solution to the LQG_{cv} problem exists for the problem imposed by the values (σ_a, μ_a) but an infinite number of solutions exist for the values (σ_b, μ_b).

The key to progress beyond the Bryson rule (1.4) is to provide a relationship between the *actual* variances Ey_i^2 , Eu_i^2 and the choices of Q_{ii} and R_{ii} in (1.5). This step is provided in Section 2 and is based upon the cost decomposition results of [6]. An iterative algorithm based upon these results is given in Section 3. Section 4 applies the algorithm to a Large Space Structure.

2.0 INPUT/OUTPUT COST ANALYSIS

Under the assumptions (2.1)

$$\left.\begin{array}{ll} (A,\ B)\ ,\ \ (A,\ D) & \text{stabilizable pairs} \\ (A,\ C)\ ,\ \ (A,\ M) & \text{detectable pairs} \end{array}\right\} \tag{2.1}$$

the LQG problem (1.1) (1.2) has the solution [ ]

$$u = G\hat{x} \quad , \quad G = -R^{-1}B^TK\hat{x} \tag{2.2a}$$

$$KA + A^TK - KBR^{-1}B^TK + C^TQC = 0 \tag{2.2b}$$

$$\dot{\hat{x}} = (A+BG)\hat{x} + F(z-M\hat{x}) \quad , \quad F = PM^TV^{-1} \tag{2.2c}$$

$$PA^T+AP - PM^TV^{-1}MP + DWD^T = 0 \tag{2.2d}$$

yielding the closed-loop regulator

$$\begin{aligned} \begin{pmatrix} \dot{x} \\ \dot{\hat{x}} \end{pmatrix} &= \begin{bmatrix} A & BG \\ FM & A+BG-FM \end{bmatrix} \begin{pmatrix} x \\ \hat{x} \end{pmatrix} + \begin{bmatrix} D & 0 \\ 0 & F \end{bmatrix} \begin{pmatrix} w \\ v \end{pmatrix} \\ y \triangleq \begin{pmatrix} y \\ u \end{pmatrix} &= \begin{bmatrix} C & 0 \\ 0 & G \end{bmatrix} \begin{pmatrix} x \\ \hat{x} \end{pmatrix} \end{aligned} \tag{2.3}$$

System (2.3) is a linear system driven by white noise and with output y. It has been established in [6] that such system outputs y_i satisfy a "cost-decomposition" property

$$V = \sum_{i=1}^{k+m} V_i^y = \lim_{t\to\infty} E\|y\|_Q^2 \tag{2.4}$$

where

$$V^y_i \triangleq \lim_{t\to\infty} \frac{1}{2} E\left(\frac{\partial \|y\|^2_Q}{\partial y_i} y_i\right) \tag{2.5}$$

The calculation and study of the "output costs" V^y_i is called "Output Cost Analysis" in [6]. We momentarily postpone the calculation of V^y_i.

Now the choice $Q = I$ in (2.5) leads to

$$V^y_i \triangleq \lim_{t\to\infty} \frac{1}{2} E\left(\frac{\partial \|y\|^2}{\partial y_i} y_i\right) = \lim_{t\to\infty} E(y_i^2) \quad i=1, \ldots, k \tag{2.6}$$

$$V^u_i \triangleq \lim_{t\to\infty} \frac{1}{2} E\left(\frac{\partial \|y\|^2}{\partial y_j} y_j\right) = \lim_{t\to\infty} E(u_i^2) \ , \quad j=k+i \quad i=1, \ldots, m \tag{2.7}$$

and for $Q = I$, note that (2.4) becomes the *unweighted* cost V°

$$V^\circ = \lim_{t\to\infty} E(\|y\|^2 + \|u\|^2) = \sum_{i=1}^{k} V^y_i + \sum_{j=1}^{m} V^u_j \ . \tag{2.8}$$

The system (2.3) evaluated according to (2.8) yields the following output cost analysis, using the output cost formula given by Eq. (4.14) of [6].

$$V^y_i = \lim_{t\to\infty} E(y_i^2) = \|c_i\|^2_X = c_i^T X c_i \tag{2.9a}$$

$$V^u_i = \lim_{t\to\infty} E(u_i^2) = R_{ii}^{-2} \|b_i\|^2_K = R_{ii}^{-2} b_i^T K b_i \tag{2.9b}$$

where X and K satisfy

$$X = \hat{X} + P \quad , \quad PA^T + AP - PM^T V^{-1} MP + DWD^T = 0 \tag{2.10}$$

$$K = K\hat{X}K \quad , \quad KA + A^T K - KBR^{-1} B^T K + C^T QC = 0 \tag{2.11}$$

$$G = -R^{-1} B^T K, \quad \hat{X}(A+BG)^T + (A+BG)^T \hat{X} + FVF^T = 0 \tag{2.12}$$

$$F = PM^T V^{-1}$$

These "output costs" V^y_i and "input costs" V^u_i provided by (2.9) show the *actual* contributions of y_i and u_i as opposed to the *desired* contributions σ_i and μ_i.

These results now allow a precise statement of the Q, R selection problem. Note that the weights Q and R appear in the cost (1.1) which is to be minimized to obtain the linear controller (2.2), but the weights do not appear in the cost (2.6)-(2.8) used for *evaluation* of the controller. Thus, using (2.9)-(2.12) the LQG_{cv} problem reduces to

The LQG_{cv} Problem:

Find Q, R *such that*

$$\|c_i\|_X^2 \leq \sigma_i^2 \tag{2.13a}$$

$$R_{ii}^{-2}\ \|b_i\|_K^2 \leq \mu_i^2 \tag{2.13b}$$

where X *and* K *are given by* (2.10)-(2.12).

3.0 AN ITERATIVE ALGORITHM FOR THE LQC_{cv} PROBLEM

The parameter optimization problem posed by (2.13) requires the simultaneous solution of $(n^2+n+2k+2m)$ algebraic equations. Gradient schemes may be developed for such purposes. However, we wish to avoid gradient schemes and the attendant numerical problems. Instead, we introduce an iterative scheme for the selection of Q_{ii}, R_{ii} such that the weighted output, input costs are constant. That is, if $\mathcal{V}_i^y(k)$ denotes the value of $\mathcal{V}_i^y$ on the k^{th} iteration of $Q_{ii}(k)$ and $R_{ii}(k)$, then

$$\mathcal{V}_i^y(k+1)\ Q_{ii}(k+1) = \mathcal{V}_i^y(k)\ Q(k) \qquad \text{for all } k \tag{3.1a}$$

and similarly

$$\mathcal{V}_i^u(k+1)\ R_{ii}(k+1) = \mathcal{V}_i^u(k)\ R_{ii}(k) \qquad \text{for all } k \tag{3.1b}$$

Now *if* these iterations converge to the maximum allowable values $(\mathcal{V}_i^y(k+1) \to \sigma_i^2$ and $\mathcal{V}_i^u(k+1) \to \mu_i^2)$ on the $(k+1)^{st}$ iteration then (3.1) reduces to

$$Q_{ii}\ (k+1) = \frac{\mathcal{V}_i^y(k)}{\sigma_i^2}\ Q_{ii}(k) = \Phi_i(k)\ Q_{ii}(k) \tag{3.2a}$$

$$R_{ii}(k+1) = \frac{\mathcal{V}_i^u(k)}{\mu_i^2}\ R_{ii}(k) = \psi_i(k)\ R_{ii}(k) \tag{3.2b}$$

where $\Phi(k)$ and $\psi(k)$ are non-negative numbers [6]. Note that the constraint (1.7) is satisfied if $\Phi(k) \leq 1$ and $\psi(k) \leq 1$. The proposed algorithm for Q, R selection is now summarized.

The LQG_{cv} Algorithm for Q, R Selection:

STEP I: Set k=0, guess an initial value for Q(0), R(0) and solve

$$PA^T+AP - PM^TV^{-1}MP + DWD^T = 0\ , \quad F = PM^TV^{-1} \tag{3.3}$$

STEP II: Compute $\mathcal{V}_i^y(k)$, $\mathcal{V}_i^u(k)$ and check for solution

$$\mathcal{V}_i^y(k) = \|c_i\|^2_{X(k)} \quad , \quad \mathcal{V}_i^u(k) = R_{ii}^{-2}(k)\ \|b_i\|^2_{\mathcal{K}} \tag{3.4}$$

where $X(k)$, $\mathcal{K}(k)$ satisfy $X(k) \triangleq \hat{X}(k) + P$, $\mathcal{K}(k) \triangleq K(k)\hat{X}(k)K(k)$ and $K(k)$, $\hat{X}(k)$ satisfy

$$0 = K(k)A+A^TK(k) - K(k)BR^{-1}(k)B^TK(k) + C^TQ(k)C \tag{3.5}$$

$$0 = \hat{X}(k)[A+BG(k)]^T + [A+BG(k)]\ \hat{X}(k) + FVF^T \tag{3.6}$$

$$G(k) = R^{-1}(k)B^TK(k) \quad , \tag{3.7}$$

STEP III: If $\Phi_i(k) \triangleq \dfrac{\mathcal{V}_i^y(k)}{\sigma_i^2} < 1$ do not change Q_{ii}

If $\psi_i(k) \triangleq \dfrac{\mathcal{V}_i^u(k)}{\mu_i^2} > 1$ do not change R_{ii}

If $\Phi_i(k) > 1$ change Q_{ii} according to

$$Q_{ii}(k+1) = \Phi_i(k)\ Q_{ii}(k) \tag{3.8}$$

If $\psi_i(k) < 1$ change R_{ii} according to

$$R_{ii}(k+1) = \psi_i(k)\ R_{ii}(k) \tag{3.9}$$

Set k=k+1 in (3.4)-(3.7) and return to STEP II until $\Phi_i(k) \leq 1$ for all i=1, ... k and $\psi_i(k) > 1$ for some i=1, ... m.

STEP IV: Change Q_{ii} according to (3.8). If $\psi_i(k) < 1$ change R_{ii} according to (3.9). Go to STEP II until $|\psi_i(k+1) - \psi_i(k)| \leq \varepsilon$ for $\psi_i(k+1) > 1$ (i.e. no more change in out of spec actuators).

There are many more choices for a Q and R algorithm other than the one posed above. To date, this algorithm has produced the best results. However, there is currently no proof that failure of this algorithm to converge means that a choice for Q and R does *not* exist. Research is continuing in this area. The algorithm does, however, attempt to use the "general" (Fig. 1) nature of LQG theory to its benefit. Specifically, choosing to adjust the r_{ii}'s for those actuators with $\psi_i(k) < 1$, using (3.2b), reduces r_{ii} and should serve to reduce $E_\infty\{y_i^2\}$ and $E_\infty\{u_i^2\}$ for actuators with $\psi_i(k) > 1$. Using (3.2b) to adjust the r_{ii}'s on actuators with $\psi_i(k) > 1$ does *not* have this *double* beneficial effect. In addition, once $\psi_i(k) \leq 1$ for all i=1, ... k, using (3.2a) to adjust q_{ii} will reduce q_{ii} which should serve to reduce $E_\infty\{u_i^2\}$ for all i=1, ... m.

4.0 HOOP-COLUMN ANTENNA EXAMPLE

The LQG_{cv} algorithm has been applied to a model for a hoop-column antenna which has 26 states, 39 measurements, 12 actuators, and 24 outputs. A detailed description of the model is presented in [6]. The initial guesses for Q and R were appropriately sized identity matricies. Figures 2, 3, and 4 are plots of $\phi_i(k)$ and $\psi_i(k)$ and reflect 14 iterations of the algorithm. The numbers appearing along the horizontal axis represent the output or actuator number. The 14 points associated with each horizontal number are the values of $\phi_i(k)$ or $\psi_i(k)$ for each iteration from 1 to 14. As was expected, Figs. 2 and 3 show that the algorithm forced all outputs to be at or below their maximum values (i.e. 1 or below on the plot). The fact that the algorithm tried but could not drive all outputs to their maximum allowable σ_i^2 is mathematically predictable but can not be discussed at this time. It is, however directly related to the sensitivity of $\hat{X}$ to changes in q_{ii}.

Figure 4 indicates that $E_\infty\{u_i^2\}$ for all actuators converged to levels above μ_i^2, and the algorithm does *not* provide a solution. However, a solution is obtained if μ_i^2 is changed to the converged value of $E_\infty\{u_i^2\}$. As mentioned earlier, the failure of the LQG_{cv} algorithm to converge has yet to be proven as sufficient for no choice of Q and R to exist.

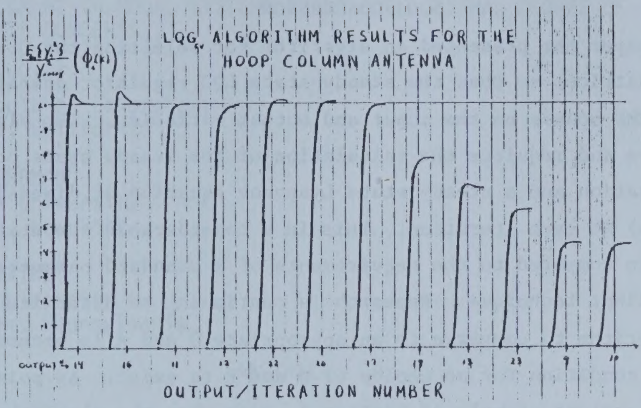


Fig. 2 Output results for the LQG_{cv} algorithm

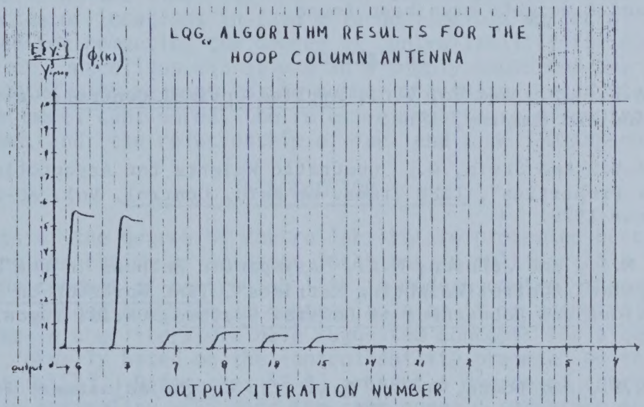


Fig. 3 Output results for the LQG_{cv} algorithm

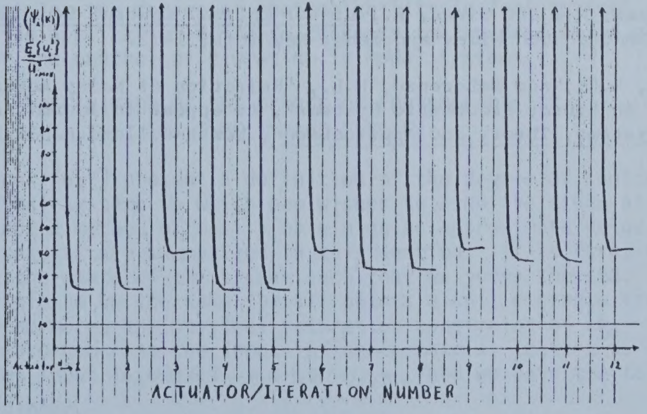


Fig. 4 Actuator results for the LQG_{cv} algorithm

5.0 CONCLUSION

This paper has presented an algorithm for selecting the Q and R weighting matrices so that the steady-state LQG regulator operates within specified RMS bounds on the input and output variables. The algorithm is iterative and requires the calculation of the steady state control Ricatti equation and a steady state Lyapunov equation of dimension n (i.e. # of states) at each iteration. This is a considerable computational savings when compared to the requirements of a standard parameter optimization gradient technique. Research is continuing to determine if failure of the algorithm to converge to an appropriate Q and R is necessary and sufficient condition for no choice of Q and R to exist. As pointed out by the hoop-column example of Section 4, even when the algorithm does not determine the required Q and R it provides the RMS regulator specifications for which Q and R weights have been found.

REFERENCES

1. Bryson, A.E., Jr., and Ho, Y.C., Applied Optimal Control, Waltham, Mass., Ginn and Company, 1969.

2. Harvey, A.H., and Stein, G., "Quadratic Weights for Asymptotic Regulator Properties", IEEE Trans. on Auto. Control, Vol. AC-23, No. 3, June 1978.

3. Grimble, M.J., and Johnson, M.A., "Asymptotic Methods in the Selectio of LQP Optimal Control Weighting Matrices", IFAC Workshop on Singular Perturbations and Robustness of Control Systems, Ohrid, Yugoslavia, July 1982.

4. McEwen, R.S., and Looze, D.P., "Quadratic Weight Adjustment for the Enhancement of Feedback Properties", American Control Conference, Arlington, Va., June 1982.

5. Kwakernaak, H., and Sivan, R., Linear Optimal Control Systems, Wiley-Interscience, John Wiley and Sons, New York, 1972.

6. Skelton, R.E., and DeLorenzo, M.L., "Selection of Noisy Actuators and Sensors in Linear Stochastic Systems", to appear in Journal of Large Scale Systems, Theory and Applications, North-Holland Publishing Company.

ACTUATOR PLACEMENT IN MODAL SYSTEMS USING FUEL OPTIMAL DEGREE OF CONTROLLABILITY CONCEPTS

R. A. LASKIN[1]
Jet Propulsion Laboratory
Pasadena, California

R. W. LONGMAN[2]
Columbia University
New York, New York

P. W. LIKINS[3]
Lehigh University
Bethlahem, Pennsylvania

ABSTRACT

Definitions of the degree of controllability relating to fuel optimal systems are given, including a bounded time definition and an alternative concept penalizing the time required. For use in problems such as the choice of actuator locations in future large flexible spacecraft, equations are developed for computing the degree of controllability for modal systems. The example of controlling vibrations in a simply supported beam is considered and studied with respect to the number of actuators used, the number of vibration modes considered, the relative importance of controlling the various modes, and the relationship of fuel and time limitations.

INTRODUCTION

Concepts of the degree of controllability are presented here for purposes of guiding the choice of actuator locations in multivariable control systems. The concept of controllability in modern control theory is a binary one -- either a system is controllable or it is uncontrollable. Several works in the literature have suggested concepts for a degree of controllability, usually based on the controllability grammian or the controllability matrix [1]. References [2-4] present a degree of controllability based on time optimality, [5] uses the same overall structure to present energy optimal degree of controllability, and in [6,7] fuel optimal concepts are considered. This work summarizes the results of [6], presents an alternative concept for the degree of controllability in fuel optimal systems, and studies the application of these concepts to the problem of placing actuators for the control of vibration in modal systems.

DEFINITION OF THE DEGREE OF CONTROLLABILITY FOR TIME T* AND FUEL F*

References [6,7] present a definition of the degree of controllability for fuel optimal systems in which one wishes to keep the state at the origin, and it develops a method of easily obtaining an approximate value of this D.O.C. The results are summarized in this section. It is first necessary to normalize the state of the system such that we judge the need for recovery to the origin to be the same for all points a unit distance from the origin. Furthermore, the physical saturation constraint of $\pm k_\beta$ for the actuator is absorbed into the input influence matrix, leaving us with normalized control input $|u_\beta(t)| \leq 1$, $\beta = 1, 2, \ldots, m$. Then we have the following:

---

[1] Senior Engineer.
[2] Professor of Mechanical Engineering.
[3] President.

Definition 1. The recovery region for time T^* *and fuel* F^* *for the normalized system (1) is the set*

$$\dot{x}(t) = Ax(t) + Bu(t) \tag{1}$$

$$\mathcal{R}_1(t) = \{x(0) \mid \exists\, u(t),\ t \in [0,T^*],\ |u_\beta(t)| \leq 1,\ \beta = 1,2,\dots,m,\ \sum_{\beta=1}^{m} \int_0^{T^*} k_\beta |u_\beta(t)|\, dt \leq F^*;\ \ni x(T^*) = 0\} \tag{2}$$ +

Definition 2. The degree of controllability for time T^* *and fuel* F^* *of the x=0 solution of the normalized system (1) is* $\rho_1 = \inf \|x(0)\|$, *infemum over all* $x(0) \notin \mathcal{R}_1$, *where* $\|\cdot\|$ *represents the Euclidean norm.*

Note that the D.O.C. definition depends on two parameters T^* and F^*. Two important limiting cases exist. When $F^* \geqslant T^* \sum_{\beta=1}^{m} k_\beta$ there is more fuel available than can be used in time T^* and this situation will be called the time optimal degree of controllability, as discussed in [2,3]. When $T^* \to \infty$ the resulting definition will be called the fuel optimal degree of controlability.

Let A have distinct eigenvalues and be diagonalized by $Q^TAP = \Lambda$ where $Q^T = P^{-1}$ and let the column partitions be denoted by $P = [p_1\, p_2 \cdots p_n]$, $Q = [q_1\, q_2 \cdots q_n]$, $B = [b_1\, b_2 \cdots b_m]$. Define $f_{j\beta}(t) = \exp(-\lambda_j t)\, q_j^T b_\beta$; $j = 1,2,\dots,n$; $\beta = 1,2,\dots,m$; $\Lambda = \mathrm{diag}(\lambda_1, \lambda_2, \dots, \lambda_n)$. Define the dead zone function $a = \mathrm{dez}(b)$ to mean $a = 0$ if $|b| < 1$, $a = \mathrm{sgn}(b)$ if $|b| > 1$, and $0 \leq a \leq 1$ if $b = +1$, $-1 \leq a \leq 0$ if $b = -1$. Also, let $\gamma_j = \sum_{\beta=1}^{m} k_\beta |\mathrm{sgn}(q_j^T b_\beta)|$ with $\mathrm{sgn}(0) \triangleq 0$. The following theorems for computing an approximate degree of controllability are proved in [6,7].

Theorem 1. Let A *have distinct eigenvalues. Then* $\mathcal{R}_1$ *can be approximated by the possibly degenerate* n *dimensional parallelepiped*

$$\mathcal{R}_1^* = \{x \mid x = \sum_{i=1}^{n} c_i \nu_i \ \forall\, c_i \ni |c_i| \leq 1\}$$

The approximation is such that $\mathcal{R}_1 \subset \mathcal{R}_1^*$, *and at least one point on every face of the parallelepiped* $\mathcal{R}_1^*$ *is contained in* $\mathcal{R}_1$. *The semiaxes* ν_j *for* λ_j *real (i.e.* $j \in \mathcal{C}$*) is*

$$\begin{aligned} \nu_j &= \Big(\sum_{\beta=1}^{m} \int_0^{T^*} |f_{j\beta}(t)|\, dt\Big) p_j \qquad \text{for } F^* \geqslant \gamma_j T^* \\ &= \Big(\sum_{\beta=1}^{m} \int_0^{T^*} f_{j\beta}\, \mathrm{dez}[f_{j\beta}/k_\beta \sigma_j]\, dt\Big) p_j \quad \text{for } F^* < \gamma_j T^* \end{aligned} \tag{3}$$

$$\sum_{\beta=1}^{m} \int_0^{T^*} k_\beta |\mathrm{dez}[f_{j\beta}/k_\beta \sigma_j]|\, dt = F^* \tag{4}$$

where the final equation identifies the value of σ_j. *For complex conjugate pairs of eigenvalues (*$j \in \mathcal{C}^*$*), the two semiaxes* ν_j^R *and* ν_j^I *are assigned. For* ν_j^R, *multiply the right hand side of (3) by 2 and substitute* $\mathrm{Re}(f_{j\beta})$, $p_j^R = \mathrm{Re}(p_j)$, σ_j^R *for* $f_{j\beta}$, p_j, *and* σ_j *in (3) and (4). Analogous substitutions apply for* ν_j^I.

Theorem 2. Let $F = [\nu_1\, \nu_2 \cdots \nu_n]$ *be nonsingular, and define the column vectors* μ_i *as the partitions of* $F^{-1} = [\mu_1\, \mu_2 \cdots \mu_n]^T$. *Then the approximate degree of controllability* $\rho_1^* = \min_i (1/\|\mu_i\|)$, *associated with the approximate recovery region* $\mathcal{R}_1^*$, *is zero if and only if the actual degree of controllability* $\rho_1 = 0$, *i.e. if and only if the system is uncontrollable.*

The way in which the vectors p_j must be altered in generating the ν_j when A has repeated eigenvalues is treated in [7], and shown to have the property described in Theorem 2.

+Note that in [6] the factors k_β were not introduced in the F^* contraint.

AN ALTERNATIVE DEFINITION OF D.O.C. FOR FUEL OPTIMAL SYSTEMS

A basic difficulty of fuel optimality is that there can exist initial conditions for which no fuel optimal solution exists if the terminal time is left free (for example, in the fuel optimal control of a double integral plant). The degree of controllability definition of the previous section avoids this difficulty by keying it to a specified time T^* and a fuel limitation F^*. An alternative approach is to leave the time free, but include a penalty in the cost functional:

Definition 3. The recovery region for combined time and fuel cost J^* *for the normalized system (1) is the set*

$$\mathcal{R}_2 = \{ x(0) \mid \exists\, u(t),\ t \in [0,T] \ni x(T) = 0 \text{ and } |u_\beta(t)| \le 1,$$
$$\beta = 1,2,\ldots,m;\ \int_0^T (k + \textstyle\sum_{\beta=1}^m k_\beta |u_\beta(t)|) dt \le J^* \}$$

Definition 4. The degree of controllability for combined time and fuel cost J^* *of the x=0 solution of the normalized system (1) is*

$$\rho_2 = \inf \| x(0) \| \quad \forall \quad x(0) \notin \mathcal{R}_2$$

where $\| \cdot \|$ *represents the Euclidean norm.*

The following theorem allows one to obtain an approximate recovery region $\mathcal{R}_2^*$, and from it an approximate degree of controllability ρ_2^*.

Theorem 3. Let A *have all distinct eigenvalues* $\lambda_1, \lambda_2, \ldots, \lambda_n$. *The recovery region* $\mathcal{R}_2$ *can be approximated by the possibly degenerate* n *dimensional parallelepiped*

$$\mathcal{R}_2^* = \{ x \mid x = \textstyle\sum_{i=1}^n c_i \nu_i \quad \forall \quad c_i \ni |c_i| \le 1 \}$$

The approximation is such that $\mathcal{R}_2 \subset \mathcal{R}_2^*$, *and at least one point on every face of the parallelepiped* $\mathcal{R}_2^*$ *is contained in* $\mathcal{R}_2$. *The semiaxes* ν_j *for* λ_j *real are given as follows*

$$\nu_j = \left(\textstyle\sum_{\beta=1}^m \int_0^T f_{j\beta}\, \mathrm{dez}\, [f_{j\beta} / k_\beta \sigma_j]\, dt \right) p_j \tag{5}$$

$$\textstyle\sum_{\beta=1}^m \{ (|f_{j\beta}(T)| - k_\beta \sigma_j) \, | \mathrm{dez} [f_{j\beta} / k_\beta \sigma_j] | \} - k \sigma_j = 0 \tag{6}$$

$$\int_0^T (k + \textstyle\sum_{\beta=1}^m k_\beta \, | \mathrm{dez} [f_{j\beta} / k_\beta \sigma_j] |) dt = J^* \tag{7}$$

where the second two equations identify the values of T *and* σ_j. *For complex conjugate pairs of eigenvectors, two semiaxes* ν_j^R *and* ν_j^I *are assigned. For* ν_j^R, *multiply the above formula for* ν_j *by two and substitute* $\mathrm{Re}(f_{j\beta})$, p_j^R, σ_j^R *for* $f_{j\beta}$, p_j, *and* σ_j *respectively. Analogous substitutions apply for* ν_j^I.

Proof. The initial state which is sent to the origin by control $u(t)$ is

$$x(0) = - \int_0^T e^{-At} B\, u(t)\, dt = - \textstyle\sum_{j=1}^n [\int_0^T e^{-\lambda_j t} q_j^T B\, u(t) dt] \, p_j$$

$$= -\sum_{j\in\mathcal{C}}\left(\sum_{\beta=1}^{m}\int_0^T f_{j\beta}u_\beta\,dt\right)p_j - \sum_{j\in\mathcal{C}^*} 2\left(\sum_{\beta=1}^{m}\int_0^T \mathrm{Re}(f_{j\beta})u_\beta\,dt\right)p_j^R$$

$$+ \sum_{j\in\mathcal{C}^*} 2\left(\sum_{\beta=1}^{m}\int_0^T \mathcal{Im}(f_{j\beta})u_\beta\,dt\right)p_j^I$$

Consider λ_j real. The control that maximizes the coefficient of p_j is one which maximizes over $u_\beta(\cdot)$ the quantity

$$l_j = \sum_{\beta=1}^{m}\int_0^T f_{j\beta}(t)\,u_\beta(t)\,dt$$

subject to $\dot{y}(t) = k + \sum_{\beta=1}^{m} k_\beta\,|u_\beta(t)|$; $y(0) = 0$, $y(T) = J^*$; $|u_\beta(t)| \le 1$, $t \in [0,T]$. Forming the Hamiltonian for this problem and maximizing over $u_\beta(t)$ gives

$$u_\beta(t) = \mathrm{sgn}\,[f_{j\beta}(t)] \text{ for } \sigma_j \le 0 \quad ; \quad u_\beta(t) = \mathrm{dez}\,[f_{j\beta}(t)/k_\beta\sigma_j] \text{ for } \sigma_j > 0$$

where σ_j is the constant costate variable for the problem. Since the time T is free, the Hamiltonian at the terminal time must be zero

$$H(T) = \sum_{\beta=1}^{m}\left(f_{j\beta}(T)\,u_\beta(T) - \sigma_j k_\beta\,|u_\beta(T)|\right) - \sigma_j k = 0$$

The first form of $u_\beta(t)$ cannot satisfy this equation except for $\sigma_j = 0$ which is obtainable as a limiting case using the second form. Hence, using the second form of $u_\beta(t)$ gives the desired ν_j of the theorem. The proof for complex conjugate pairs of roots is analogous.

If each of the integral coefficients in $x(0)$ could be manipulated independently over the admissible controls, then the resulting set of $x(0)$ would describe $\mathcal{R}_2^*$. Since we are restricted to one control $u(t)$ for all coefficients $\mathcal{R}_2 \subset \mathcal{R}_2^*$, and by construction a control has been generated which makes at least one point on each face of $\mathcal{R}_2^*$ an element of $\mathcal{R}_2$. ∎

Having developed a parallelepiped approximation for the recovery region $\mathcal{R}_2$, an approximate degree of controllability ρ_2^* can be generated as with the previous definition (Theorem 2).

MODAL EQUATIONS

This section is devoted to finding the approximate degrees of controllability ρ_1^* and ρ_2^* for modal systems

$$\ddot{\eta}_j + 2\zeta_j\omega_j\dot{\eta}_j + \omega_j^2\eta_j = \Gamma_j^T u \quad ; \quad j = 1, 2, \ldots, n/2 \qquad (8)$$

We consider ρ_1^* first. More detail can be found in [7], including the influence of rigid body modes. The Γ_j are the coefficients associated with normalized controls, $|u_\beta| \le 1$, so that they contain the actuator strength k_β multiplying the β^{th} element. The symbol $\overline{\overline{\Gamma}}_j$ will be used when this factor is absent. The normalization required in forming equation (1) will be taken as N_j for the modal variables η_j , and for lightly damped systems or ones without damping, $\dot{\eta}_j$ will have an amplitude approximately ω_j times that of η_j . Thus the state variables $x_{2j-1} = \eta_j / N_j$, $x_{2j} = \dot{\eta}_j / \omega_j N_j$ will be assigned. Computation of all of the quantities needed in the complex conjugate version of equations (3) and (4) gives the approximate degree of controllability for (8) as

$$\rho_1^* = \min_j\left[l_j \sqrt{(1-\zeta_j^2)/(1+\zeta_j^2)}\right] \qquad (9)$$

$$\ell_j = 2\int_0^{T^*}\sum_{\beta=1}^m \Psi_{j\beta} k_\beta e^{\omega_j \zeta_j t}\cos(\omega_j\sqrt{1-\zeta_j^2}\,t)\,\mathrm{dez}\left(\Psi_{j\beta}e^{\omega_j\zeta_j t}\cos(\omega_j\sqrt{1-\zeta_j^2}\,t)/\sigma_j\right)dt \tag{10}$$

$$\Psi_{j\beta} = \Gamma_{j\beta}/(2\omega_j N_j\sqrt{1-\zeta_j^2}\,k_\beta)$$

$$\sum_{\beta=1}^m \int_0^{T^*} k_\beta \left|\mathrm{dez}\left(\Psi_{j\beta}e^{\omega_j\zeta_j t}\cos(\omega_j\sqrt{1-\zeta_j^2}\,t)/\sigma_j\right)\right| dt = F^* \tag{11}$$

The computaion of ℓ_j requires knowledge of σ_j (which can be considered σ_j^R or σ_j^I) from the last equation above, which can be obtained analytically in certain cases, but in general requires a simple scan over the positive values for σ_j using a computer.

For the no damping case the dead zone function first switches values from 1 to 0 at time $t_\beta = (1/\omega_j)\cos^{-1}(\sigma_j/|\Psi_{j\beta}|)$ if $\sigma_j \leq |\Psi_{j\beta}|$ and is 0 from time $t_\beta = 0$ if $\sigma_j > |\Psi_{j\beta}|$. The fuel used by the β^{th} actuator from (11) is $(2k_\beta T^*\omega_j/\pi)\,t_\beta$ where the assumption $T^* >> 2\pi/\omega_j$ has been used to justify neglecting a small correction due to any partial completion of the last period of oscillation of $\cos\omega_j t$. This assumption is used throughout the sequel. Equation (11) then takes on the far simpler form $(2T^*/\pi)\sum_{\beta=1}^m k_\beta\cdot\cos^{-1}(\sigma_j/|\Psi_{j\beta}|) = F^*$ where the inverse cosine is to be taken as zero if its argument exceeds unity. Once σ_j is found from this equation, the ℓ_j of (10) can be computed under the above assumption by recognizing the integral over $[0, T^*]$ is very nearly equal to $4T^*/(2\pi/\omega_j)$ times the integral over $[0, t_\beta]$. This results in the following simple formula for the approximate degree of controllability for time T^* and fuel F^* for system (8)

$$\rho_1^* = \min_j \ell_j = \min_j\left[(4T^*/\pi)\sum_{\beta=1}^m k_\beta(\Psi_{j\beta}^2 - \sigma_j^2)^{1/2}\right] \tag{12}$$

where the square root is to be taken as zero if the radical is negative.

It is interesting to study the limiting cases of fuel optimal and time optimal degree of controllability. Denote the weighted p-norm of a vector v by $\|v\|_{p,w} = (\sum_{i=1}^n |v_i|^p w_i)^{1/p}$ where the w_i are positive constants. As $p\to\infty$, the weighting factors become superfluous and we can write $\|v\|_\infty = \max_i |v_i|$. For the time optimal case $\sigma_j \to 0$ in the above equation producing the result below which was given in [2]. The fuel optimal limiting case can be solved by letting $T^*\to\infty$, and noting that the σ_j equation gives $\sigma_j \to \max_\beta|\Psi_{j\beta}|$. The physical interpretation is that as concern about response time recedes, only the most favorably located actuator (with respect to the j^{th} mode) will be used in order to conserve fuel. Mathematically this causes the summation to dissappear in the equation for σ_j making it solvable analytically. Reference [5] develops an energy optimal degree of controllability using $\int_0^{T^*} u^T R u\,dt = E^*$. For undamped modal systems and using $R^{-1} = \mathrm{diag}(k_1, k_2, \ldots, k_m)$, it also takes on a simple form. These three cases are summarized as

Time Optimal: $$\rho^* = (2T^*/\pi)\min_j(\|\bar{\Gamma}_j\|_{1,k}/\omega_j N_j) \tag{13}$$

Fuel Optimal: $$\rho^* = F^*\min_j(\|\bar{\Gamma}_j\|_\infty/\omega_j N_j) \tag{14}$$

Energy Optimal: $$\rho^* = \sqrt{E^*T^*}\min_j(\|\bar{\Gamma}_j\|_{2,k}/\omega_j N_j) \tag{15}$$

It is particularly appealing to have simple expressions for these concepts of the degree of controllability in terms of weighted norms of the rows of the input influence matrix. The $\bar{\Gamma}_j$ are the original rows of this matrix before the saturation limits k_β have been introduced.

Now consider the computation of ρ_2^* for undamped systems assuming $T >> 2\pi/\omega_j$ as before. Equations (5-7) for complex conjugate roots give

$$\rho_2^* = \min_j\left[(4T/\pi)\sum_\beta(\Psi_{j\beta}^2 - \sigma_j^2)^{1/2}\right] \tag{16}$$

$$\Sigma_\beta \{ (|\Psi_{j\beta} \cos \omega_j T| - k_\beta \sigma_j) |\mathrm{dez}(\Psi_{j\beta} \cos \omega_j T/\sigma_j)| \} - k\sigma_j = 0 \tag{17}$$

$$T = J^* / [k + (2/\pi) \Sigma_\beta k_\beta \cos^{-1}(\sigma_j / |\Psi_{j\beta}|)] \tag{18}$$

The summations on β are only over those β for which $|\Psi_{j\beta}| \geqslant \sigma_j$. Substitution of (18) in (17) reduces the problem to the same type of numerical scanning problem for σ_j as before. Again, the limiting cases are illuminating. Consider $k \gg \Sigma_{\beta=1}^{m} k_\beta$ which should correspond to the time optimal case. Equation (18) indicates that $T \rightarrow J^*/k$, and (17) that σ_j must be small compared to $|\Psi_{j\beta}|$, which leads to $\rho_2^* = (2J^*/\pi k) \min_j [\|\overline{\Gamma}_j\|_{1,k}/\omega_j N_j]$. Since J^*/k is equivalent to T^* this expression is the time optimal D.O.C. When $k \rightarrow 0$, $\sigma_j \rightarrow \max_\beta |\Psi_{j\beta}|$ giving $\rho_2^* = J^* \min_j [\|\overline{\Gamma}_j\|_\infty/\omega_j N_j]$ which is the fuel optimal D.O.C. solution since $k = 0$ implies $J^* = F^*$.

We conclude that the use of the degree of controllability in Definition 4 is only slightly more difficult than Definition 3 for undamped modal systems, and gives strikingly similar results. The choice for the use of one definition over the other must lie in the ability to prescribe the relevant parameters (T^* and F^* versus J^* and k) in any given physical problem. In most situations one expects T^* and F^* to be the more natural parameters to prescribe.

EXAMPLES

Examples of actuator placement on a unit lenght simply supported Euler-Bernoulli beam are given to study the behavior of the degree of controllability for time T^* and fuel F^*. In this case, $\overline{\Gamma}_j^T = [k_1 \sin j\pi\bar{x}_1, \ldots, k_m \sin j\pi\bar{x}_m]$ where $\bar{x}_\beta$ is the location of the β^{th} force actuator, and $\omega_j = (j\pi)^2$, where the physical constants of the beam have been chosen for simplicity of these equations (i.e. $(EI/M)^{1/2}/L^2 = 1$). The actuators are considered gas driven reaction jets for which fuel optimality in the placement of actuators is natural. Figure 1 shows the optimal actuator location as a function of the modal weighting factors when there are two modes being controlled. When $N_2 \gg N_1$, the actuator is placed at the maximum of the second mode (.25), and when $N_1 \gg N_2$, the actuator only approaches the maximum point for the first mode (.50) and fails to reach it since the second mode is uncontrollable at .50. It should be pointed out that in the single actuator case one can obtain an analytic solution

$$\rho_1^* = (2k_1 T^*/\pi) \sin(\pi F^*/2k_1 T^*) \min_j (|\sin j\pi\bar{x}_1|/\omega_j N_j) \tag{19}$$

for the approximate degree of controllability, and its form indicates that the optimal actuator locations are independent of k_1, T^*, and F^*.

The interest in controlling different modes will decrease with frequency since it is much harder to obtain significant amplitudes from disturbances to higher frequency modes. If it is assumed that the modes have equal energies then the amplitudes decrease with frequency producing a weighting in the form $N_j = 1/\omega_j$. One expects that the energies in the modes will actually decrease with frequency so that a weighting of the form $N_j = 1/\omega_j^2$ might be more appropriate. Figure 2 studies the optimal actuator location for both weighting schemes; the former has the actuator tending to an end of the beam as the number of modes considered increases, whereas the second weighting scheme does not. Plots of the degree of controllability as a function of actuator location using the first weighting scheme show that the central section of the beam has a more and more jagged appearance as more modes are introduced (see [7] for plots), and it is this behavior that pushes the actuator locations toward the ends as the number of modes increases. The first weighting scheme is used in the remaining plots.

Figures have also been generated (see [7]) to study the usefulness of using stronger actuators when the fuel consumption of the optimally located actuator is proportional to the actuator strength. It was found that the degree of controllability saturates as a function of this strength at a value dependent on the total fuel allotment F^*(what is plotted is $\pi\rho^*/[2T^* \cdot \min_j(|\sin j\pi\bar{x}_1|/\omega_j N_j)]$ which can be thought of as proportional to ρ^* for fixed T^* once $\bar{x}_1$ is chosen optimally). Appearantly, if the total fuel available is too limited there is no way to improve the performance using a stronger actuator. By contrast, for the time optimal case ($F^* > k_1 T^*$) the D.O.C. varies linearly with k_1. A rough physical interpretation is that in the time optimal case the control is bang-bang with the thruster essentially always working against the local beam velocity, whereas in the fuel optimal (large T^*) case the thruster is pulsed only around the time of maximum local beam velocity. Since fuel is limited in the latter case, only a certain number of finite duration pulses is permitted for T^*large but not infinite. Increasing k_1 increases the amplitude of the pulses, but cuts their duration due to the fuel constraint, and the change in the amount of energy being removed from the beam per oscillation approaches zero.

Consider the use of two actuators to control the beam. No closed form expression is available for ρ_1^*, but rather a scanning and iterative algorithm isemployed to find the actuator locations to maximize ρ_1^*. There are many local maxima and the results presented are intended to be global maxima although no guarantee is made. Figure 3 shows the way in which the fuel optimal D.O.C. ($F^*/mT^* = F^*/2T^* \rightarrow 0$) and the time optimal D.O.C. ($F^*/mT^* = 1$) differ in the desired actuator locations. In the former case the actuators are placed at distances .25 and .50 along the beam corresponding to the maxima of the two modes considered. In the latter case both actuators are placed at .333 which is the intersection of the two mode shapes. The transition between these two location sets is made in a discontinuous manner as a function of F^*/mT^*at approximately .33. As in all of the plots, the symmetry of the problem makes any actuator location and its symmetric location of equal utility. For simplicity, only the optimal locations on one side of the beam are indicated. When more modes are being controlled the results are no longer so transparent, but the general behavior remains similar (see [7]).

When one studies the variation of actuator position with actuator strength ratio k_2/k_1 one again finds step changes which interchange the position for each actuator as k_2/k_1 passes through unity, corresponding to putting the strongest actuator at the best location. The fuel optimal D.O.C. exhibits some additional interesting phenomena. Over a considerable range of k_2/k_1 using three controlled modes and two actuators, one actuator must be located at .25, but the other is free to occupy any position in a continuum of equally favorable positions. This can be understood by noting that the form of (14) shows that only one element from each row of the input influence matrix has an impact on ρ^*.

One can pose design problems which ask how many optimally located actuators of a specified type and strength would be needed to obtain a specified degree of controllability. Figure 4 shows such a design chart for fifteen controlled modes, giving both time optimal and fuel optimal degree of controllability as a function of the number of actuators. The controller does better with each additional actuator in both cases. Note that use of four actuators of unit strength placed optimally gives much better performance than one actuator of four times the strength, for both definitions of D.O.C. One expects that as the number of actuators tends to infinity, performance is further improved by the use of what is physically a distributed actuator.

REFERENCES

1. Kalman, R.E., Ho, Y.C. and Narendra, K.S., "Controllability of Linear Dynamical Systems," *Contributions to Differential Equations*, Vol. 1, No. 2, 1961, pp. 182-213.
2. Viswanathan, C.N., Longman, R.W. and Likins, P.W., "A Definition of the Degree of Controllability -- A Criterion for Actuator Placement," *Proceedings of the Second VPI&SU/AIAA Symposium on Dynamics and Control of Large Flexible Spacecraft*, Blacksburg, Va., June 1979. Updated version accepted for publication in the *Journal of Guidance, Control and Dynamics*.
3. Viswanathan, C.N. and Longman, R.W., "The Determination of the Degree of Controllability for Dynamic Systems with Repeated Eigenvalues," Proceedings of the NCKU/AAS Symposium on Engineering Science and Mechanics, Tainan, Taiwan, Dec. 1981, *Advances in the Astronautical Sciences*, to appear. Accepted for publication in *The Journal of the Astronautical Sciences*.
4. Klein, G., Lindberg, R. and Longman, R.W., "Computation of a Degree of Controllability via System Discretization," *Proceedings of the Third VPI&SU/AIAA Symposium on Dynamics and Control of Large Flexible Spacecraft*, Blacksburg, Va., June 1981. Accepted for publication in the *Journal of Guidance, Control and Dynamics*.
5. Longman, R.W. and Alfriend, K.T., "Energy Optimal Degree of Controllability and Observability for Regulator and Maneuver Problems," *Proceedings of the 16th Conference on Information Sciences and Systems*, Princeton, N.J., March 17-19, 1982.
6. Laskin, R.A., Longman, R.W. and Likins, P.W., "A Definition of the Degree of Controllability for Fuel Optimal Systems," *Proceedings of the Third VPI&SU/AIAA Symposium on Dynamics and Control of Large Flexible Spacecraft*, Blacksburg, Va., June 1981.
7. Laskin, R.A., *Aspects of the Dynamics and Controllability of Large Flexible Structures*, Ph.D. Dissertation, Columbia University, 1982.

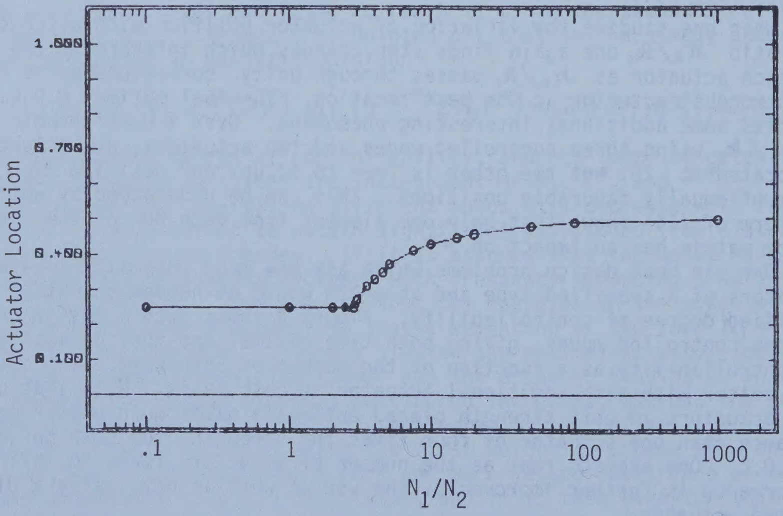


FIG. 1. Optimal Actuator Location vs. N_1/N_2 -- Two Controlled Modes, Single Actuator.

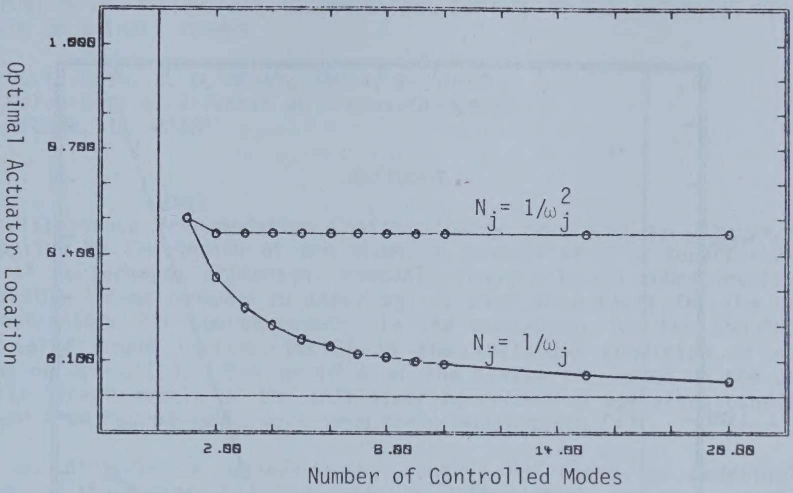


FIG. 2. Optimal Actuator Location vs. Number of Controlled Modes -- Single Actuator, Two Weighting Schemes.

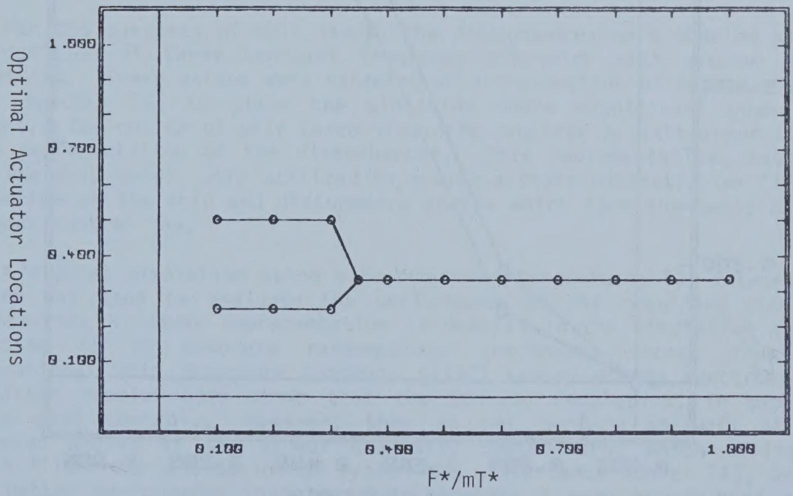


FIG. 3. Optimal Actuator Locations vs. $\dot{F}^*/mT^*$ -- Two Controlled Modes, Two Actuators.

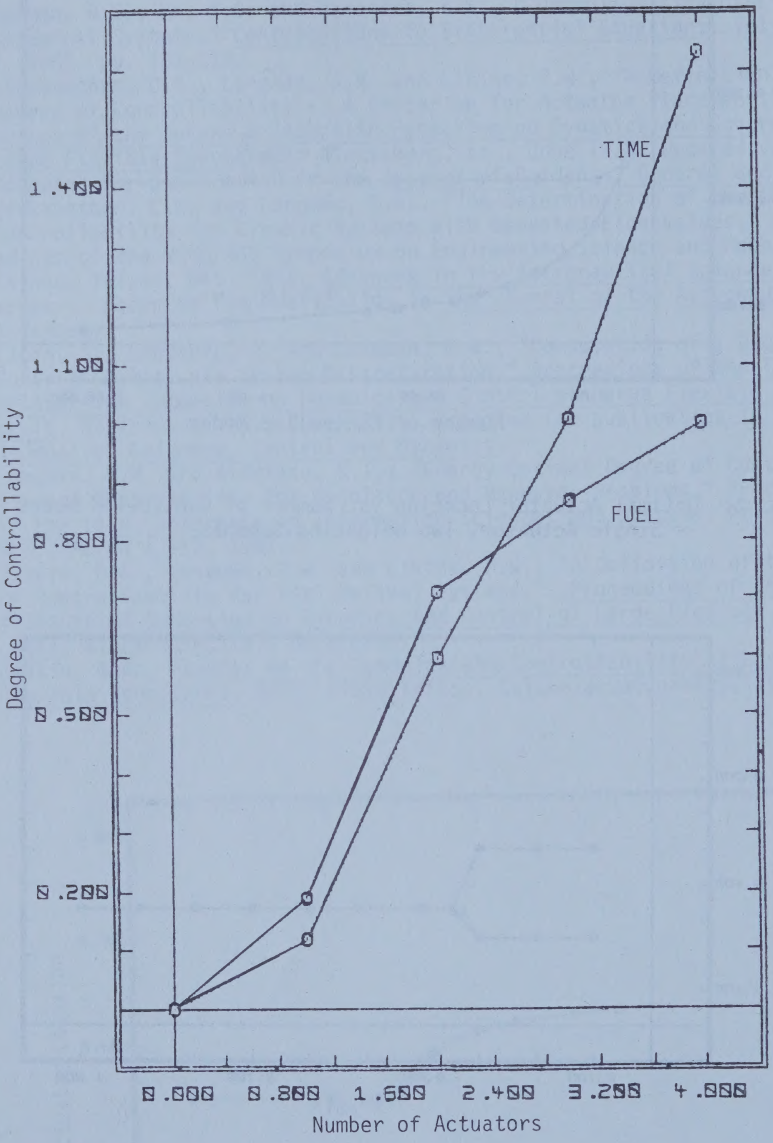


FIG. 4. Degree of Controllability vs. Number of Actuators -- Fifteen Controlled Modes, Time Optimal and Fuel Optimal.

APPLICATION OF DISTURBANCE ACCOMMODATING CONTROL TO THE AUTOMATIC STEERING CONTROL OF A LARGE TANKER

R. E. REID, B. C. MEARS, AND A. K. TUGCU
University of Illinois at Urbana-Champaign
Urbana, IL 61801

ABSTRACT

Disturbance Accommodating Control theory developed by Johnson [1-2] is applied to the design of the steering controller of a supertanker. A form of performance criterion, commonly thought to be representative of propulsion losses related to steering, is used as a basis for the design of controllers for course-keeping in the open-seas, for the 250,000 DWT Very Large Crude Carrier (VLCC) in the full-load condition at normal operating speed [3]. The dynamics of the system are based on the use of locally linear models of the ship/steering around an operating condition, derived from hydrodynamic data from scale model tests [4].

The disturbance accommodating control is based on modeling the process disturbances by developing a differential equation which the disturbances are known to satisfy. This technique, referred to as disturbance state modeling, can account for any deterministic component in the disturbances. This approach was selected for analysis because sea waves reflect some degree of regularity [5], which indicates that disturbances due to "somewhat regular" waves could be modeled deterministically, or perhaps by a combination of deterministic and stochastic approaches.

For the purposes of this study, the disturbances were modeled as the superposition of three constant frequency sinusoids with unknown phase angles [6]. These values were selected after inspection of representative power spectra [6] to place the sinusoids where significant power was present. The choice of only three sinusoids required a sixth-order state-space representation of the disturbances. This representation, together with the ship model, were utilized to design a state reconstructor [1] for estimation of the ship and disturbance states which form the basis of the feedback control law.

A digital simulation using a fourth-order Runge-Kutta [7] integration routine was used to evaluate the performance of the resulting control. The Beaufort 8 seaway representation is modeled in the simulation as the summation of 25 discrete random-phase components chosen from the International Ship Structure Congress (ISSC) seaway energy spectrum [8]. Simulation results have shown that the Johnson regulators, in general, afford good control. However, they do not perform as well at low encounter angles as the linear quadratic controllers, which utilize a Kalman filter for estimation of system and disturbance states [4], but do show better performance in beam seas [9] where a very narrow-band power spectrum is involved.

REFERENCES

[1] Johnson, C. D., "Accommodation of External Disturbances in Linear Regulator and Servomechanism Problems," Trans. on Automatic Control, Vol. AC-16, No. 6, Dec. 1971.

[2] Johnson, C. D., "Accommodation of Disturbances in Optimal Control Problems," Int. J. of Control, Vol. 15, No. 2, 1971.

[3] Reid, R. E., and B. C. Mears, "Design of the Steering Controller of a Supertanker Using Linear Quadratic Control Theory: A Feasibility Study," Trans. on Automatic Control, Vol. AC-27, No. 4, Aug. 1982.

[4] Reid, R. E., B. C. Mears, and D. E. Griffin, "Energy Losses Related to Automatic Steering of Ships: Their Evaulation and Minimization," ASME Ocean Engr. Div., 1981 Winter Annual Meeting, Washington, D.C., No. 1981.

[5] Saelid, S., and N. A. Jenssen, "Adaptive Autopilot with Wave Filter," Sixth IFAC Symp. on Identification and System Parameter Estimation, Washington, D.C., June 1982.

[6] Mears, B. C., "Design and Analysis of Automatic Steering Control Based on Non-White Disturbance Inputs for a VLCC in a Seaway," Master of Science Thesis, University of Illinois at Urbana-Champaign, Aug. 1982.

[7] Pennington, R. H., Introductory Computer Methods and Numerical Analysis, John Wiley and Sons, New York, 1972.

[8] Michel, W. H., "Sea Spectra Simplified," Marine Tech., Jan. 1968.

[9] Reid, R. E., B. C. Mears, K. A. Wise, A. K. Tugcu, D. E. Griffin, and V. E. Williams, "Comparison of Automatic Steering Performance of a VLCC in a Seaway Resulting from Application of LGC and Classical Control System Design Techniques," Proc., Sixth Ship Control Systems Symp., Ottawa, Canada, Oct. 1981.

OBSERVABILITY AND IMPULSE OBSERVERS IN DESCRIPTOR-VARIABLE SYSTEMS*

DANIEL COBB
Department of Electrical Engineering
University of Toronto
Toronto, Ontario, Canada M5S 1A4

ABSTRACT

A theory of observability for systems $E\dot{x}=Ax+Bu$, $y=Cx$, E singular, which complements that for controllability, already existing in the literature, is presented. A natural extension of the state-space duality theorem is shown to hold, and an observer, which detects impulsive behavior, is studied.

1. INTRODUCTION

In this paper we discuss generalizations of the concepts of controllability and observability to systems of the form

$$E\dot{x} = Ax + Bu \tag{1}$$

$$y = Cx \tag{2}$$

where E and A are n×n matrices, B is n×n and C is k×n. According to the development in Gantmacher [1], (1) and (2) may be decomposed (after a change of basis) according to $x = \begin{bmatrix} x_s \\ x_f \end{bmatrix}$ where

$$\dot{x}_s = A_s x_s + B_s u \tag{3}$$

$$A_f \dot{x}_f = x_f + B_f u \tag{4}$$

$$y = C_s x_s + C_f x_f \tag{5}$$

If

$$r = \deg|Es-A| \tag{6}$$

then x_s is an r-vector and A_f is nilpotent. The unique solution of (4) is

$$x_f = -\sum_{i=0}^{q-1} A_f^i B_f u^i \tag{7}$$

where q is the index of A_f and u^i denotes the ith derivative.

Inevitably, when considering such a system, one is faced with the problem of interpreting "inconsistent" initial conditions, as they are termed by Campbell [2], which may be thought of as points in the state

---

*This work was supported in part by the Natural Sciences and Engineering Research Council of Canada under grant no. A1699.

space through which no solution passes for a given u. It can be shown that, in practice, such initial values may be achieved if the system operates in the presence of disturbances. This situation might result, for example, from the addition of a disturbance input v to the right side of (1) which, in a worst case analysis, would be multiplied by the identity matrix. Hence, by assigning t=0 to the instant when the disturbance disappears, any initial condition can result. It can also be shown (see [2], for example) that inconsistent initial conditions may lead to impulses and their derivatives in the natural response. Specifically,

$$x_f = - \sum_{i=1}^{q-1} \delta^{i-1} A_f^i x_{0f} \tag{8}$$

where x_{0f} is the component of the initial condition corresponding to subsystem (4). The impulses that appear in this deterministic approach to disturbances, reflect the differentiating structure of the system as determined by (7). Analogously, in a stochastic approach, this structure would result in high frequency amplification of noise.

The first attempts at extending the concepts of controllability and observability to the system (1) were made by Rosenbrock [3] and later by Verghese et al. [4]. Theirs is a frequency domain approach, controllability and observability being described in terms of infinite input and output decoupling zeros. Our work is closer to that of Yip et al. [5], who takes a time-domain approach, but does not account for inconsistent initial conditions. This fact does not cause any significant problems when it comes to aspects of controllability, since there the reachability of states for t>0 is the key issue. Observability, on the other hand, depends heavily on initial conditions. Ignoring inconsistent initial conditions leads to a definition of observability which is not the algebraic dual of controllability. These difficulties will be dealt with in Section 2.

In Section 3 we will discuss the control and observation of the impulsive behavior of (1). This will lead to system properties which do not appear in state-space theory. The results in Section 3 will be applied in Section 4 to the problem of constructing a dynamic observer which reproduces the impulsive portion of the internal variable x, given only input and output information. Finally, we will discuss the observer's use in a feedback compensation scheme which eliminates impulsive behavior.

2. OBSERVABILITY

We begin by reviewing the definition of controllability as given in [5]. Let

$$\mathcal{R}_s = \sum_{i=0}^{r-1} \operatorname{Im}(A_s^i B_s) \tag{9}$$

$$\mathcal{R}_f = \sum_{i=0}^{q-1} \operatorname{Im}(A_f^i B_f) \tag{10}$$

$$\mathcal{R} = \mathcal{R}_s \oplus \mathcal{R}_f \tag{11}$$

$\mathcal{R}$ is the controllable subspace corresponding to (1). Let (1) have initial condition $x_0 = \begin{bmatrix} x_{0s} \\ x_{0f} \end{bmatrix}$ and let C^{q-1} be the space of q-1 times continuously differentiable functions on $[0,\infty)$.

Definition

System (1) is controllable if for every $\tau>0$ and x_0, $w \in \mathbb{R}^n$ there exists $u \in C^{q-1}$ such that $x(\tau)=w$.

Theorem 1 (Yip)

Let $\tau>0$ and x_0, $w \in \mathbb{R}^n$. There exists $u \in C^{q-1}$ such that $x(\tau)=w$ iff $w \in \mathcal{R}$.

Corollary

The system (1) is controllable iff $\mathcal{R}=\mathbb{R}^n$.

We would like to define observability so that the dual nature of controllability and observability is preserved. In order to achieve this and to account for all initial conditions, it becomes necessary to restrict the system under consideration to $[0,\infty)$. In the state-space case this would not change the form of the system equation since all solutions are continuous. However, in our case the situation is complicated by the existence of distributions.

Let x_+ and y_+ be the distributions which 1) are equal to x and y on $(0,\infty)$, 2) vanish on $(-\infty,0)$, and 3) contain at the origin the same impulses and their derivatives as x and y. Then x_+ and y_+ satisfy the equations

$$E\dot{x}_+ = Ax_+ + Bu + \delta Ex(0^-) \tag{12}$$

$$y_+ = Cx_+ \tag{13}$$

In fact, the solution of (12) is unique within the space of distributions which vanish on $(-\infty,0)$. This formulation allows us to speak of system trajectories resulting from various initial conditions without having to consider the disturbances that generate those initial conditions.

We are now in a position to define observability. Let

$$N_s = \bigcap_{i=0}^{r-1} \operatorname{Ker}(C_s A_s^i) \tag{14}$$

$$N_f = \bigcap_{i=0}^{q-1} \operatorname{Ker}(C_f A_f^i) \tag{15}$$

$$N = N_s \oplus N_f \tag{16}$$

N is the unobservable subspace corresponding to (1). Let C_p^{q-1} be the space of q-1 times piecewise continuously differentiable functions on $[0,\infty)$.

Definition

The system (1) is observable if knowledge of $u \in C_p^{q-1}$, y_+, and $y(0^-)$ are sufficient to determine $x(0^-)$.

Theorem 2

Let u=0 in (12). Then $y_+=0$, $y(0^-)=0$ iff $x(0^-) \in N$.

Corollary

The system (1) is observable iff $N=0$.

The theorem can be proven by combining (8) with the decomposition of (12) and (13). Since the system is linear, Theorem 2 is equivalent to the statement that, for any given u, two initial conditions, whose difference lies in N, give rise to the same output. Hence, the corollary follows easily. Later we will see that observability, as we have defined it, is the algebraic dual of controllability.

3. IMPULSE CONTROLLABILITY AND OBSERVABILITY

Since it is unique to systems with a singular E matrix, impulsive behavior in x warrants special attention. In this section we outline a theory which emphasizes the control and observation of impulses. To facilitate the development, we denote by $x[\tau]$ the impulsive part of x at $t=\tau$. Similarly, let $y[\tau]$ be the impulsive part of y at τ. Then

$$E\dot{x}[\tau] = Ax[\tau] - \delta_\tau E(x(\tau^+) - x(\tau^-)) \tag{17}$$

$$y[\tau] = Cx[\tau] \tag{18}$$

where δ_τ is the unit impulse at τ. It can be shown that the solution of (17) is unique within the space of distributions with point support at τ. Decomposition of (17) reveals that

$$x_f[\tau] = \sum_{i=0}^{q-1} \delta_\tau^{i-1} A_f^i (x(\tau^+) - x(\tau^-)) \tag{19}$$

Hence, x can only contain impulses of the form (19). We would like to know which impulses of the class determined by (19) can actually be generated by choosing u appropriately, and which can be detected given only output information.

Definition

System (1) is impulse controllable if for every $\tau \in \mathbb{R}$, $w \in \mathbb{R}^{n-r}$ there exists $u \in C_p^{q-1}$ such that

$$x[\tau] = \sum_{i=1}^{q-1} \delta_\tau^{i-1} A_f^i w$$

Let

$$\mathcal{I}_r = \sum_{i=1}^{q-1} \mathrm{Im}(A_f^i B_f) \tag{20}$$

We call $\mathcal{I}_r$ the impulse controllable subspace. This name is motivated by the following result.

Theorem 3

There exists $u \in C_p^{q-1}$ such that $x[\tau] = \sum_{i=1}^{q-1} \delta_\tau^{i-1} A_f^i w$ iff $\sum_{i=1}^{q-1} \delta_\tau^{i-1} A_f^i w \in \mathcal{I}_r$.

Corollary

The following statements are equivalent:

1) System (1) is impulse controllable.
2) $\mathcal{I}_r = \mathrm{Im}\, A_f$.
3) There exists an m×n matrix K such that $\deg|Es-(A+BK)| = \mathrm{rank}\, E$.

Theorem 3 can be proven in a manner similar to Theorem 2. The equivalence of conditions 2) and 3) in the corollary, was established in [6] where the problem of eliminating impulses due to inconsistent initial conditions (or, equivalently, due to arbitrary disturbances) was explored. The corollary of Theorem 3 indicates then that all impulsive transients can be eliminated with linear feedback iff they can be generated with an appropriate u.

We now consider the dual of impulse controllability.

Definition

System (1) is impulse observable if, for every $\tau \in \mathbb{R}$, knowledge of $y[\tau]$ is sufficient to determine $x[\tau]$.

Let

$$I_n = \bigcap_{i=1}^{q-1} \operatorname{Ker}(C_f A_f^i) \tag{21}$$

I_n is the impulse unobservable subspace.

Theorem 4

$y[\tau]=0$ iff $x(\tau^+)-x(\tau^-) \in I_n$.

Corollary

The following statements are equivalent:

1) System (1) is impulse observable.
2) $I_n = \operatorname{Ker} A_f$.
3) There exists an n×k matrix K such that $\deg|Es-(A+KC)| = \operatorname{rank} E$.

The equivalence of 1) and 3) can be proven by the following duality theorem. Consider the dual system

$$E'\dot{x} = A'x + C'u \tag{22}$$

$$y = B'x \tag{23}$$

Theorem 5

1) System (1) is controllable iff the dual system is observable.
2) System (1) is impulse controllable iff the dual system is impulse observable.

The proof of this theorem is somewhat complicated, involving the relationship among the subspaces R_s, R_f, N_s, etc. for the two systems. The duality result serves as partial motivation for the exact definitions which we chose for observability. In the next section we will see that there is in fact a more pragmatic justification for our choice of definitions.

4. AN IMPULSE OBSERVER

In this section we consider the problem of constructing a dynamic system which utilizes input and output information from (1) and generates impulses equal to those of x, as given by (19). In particular we are looking for an impulse observer of the form

$$E\dot{z} = Gz - Ky + Hu \tag{24}$$

such that the estimate z exhibits the same impulsive behavior as x. Hence, we would like to find values for G, K, and H so that the error

$$e = z - x \tag{25}$$

contains no impulses.

The error e is governed by

$$E\dot{e} = Gz - Ky - Ax + (H-B)u \tag{26}$$

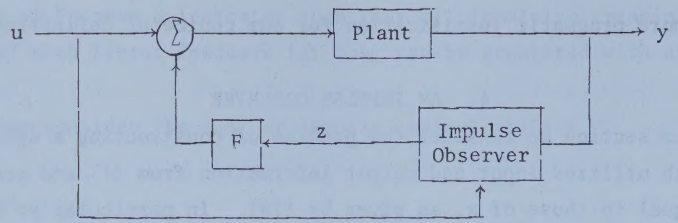
so if we choose H=B and G=A+KC then

$$E\dot{e} = (A+KC)e \tag{27}$$

From the corollary to Theorem 4, a matrix K can be found so that e has no impulses iff (1) is impulse observable.

The problem of stabilizing the observer still needs to be addressed. That this can be accomplished iff subsystem (3) is observable, follows from Theorem 5 and the results of [6]. Under mild conditions then a stable impulse observer can be constructed for the system (1) which identifies internal high frequency behavior in (1), due to random disturbances.

The results of [6], which deal with elimination of impulses via linear feedback when x is available, can now be augmented to handle the case where only y is available. To this end we consider the compensation scheme



where the plant is characterized by (1). The closed loop system may be modelled in terms of e and x yielding

$$\begin{bmatrix} E & 0 \\ 0 & E \end{bmatrix}\begin{bmatrix}\dot{x}\\ \dot{e}\end{bmatrix} = \begin{bmatrix} A+BF & BF \\ 0 & A+KC\end{bmatrix}\begin{bmatrix}x\\ e\end{bmatrix} + \begin{bmatrix}B\\ 0\end{bmatrix}u \tag{28}$$

since u=Fz=Fx+Fe. Because of the triangular structure of (28), and the results of [6] and this paper, impulse controllability and impulse observability of (1) are necessary and sufficient to guarantee that (28) exhibits no impulsive behavior whatsoever. The eigenvalues of the closed-loop system are simply those of the plant and observer individually. They can be placed arbitrarily iff subsystem (3) is both controllable and observable.

5. CONCLUSIONS

The key to understanding the dual nature of controllability and observability clearly lies in the observation that all initial conditions may occur if the system operates in the presence of disturbances. This idea has led us to a natural extension of the duality theorem from state-space theory. We have also seen that, being unique to systems with a singular E matrix, impulsive behavior motivates the study of certain aspects of controllability and observability which do not appear when E is nonsingular. Finally, the problem of reorganizing system structure by applying feedback, so that internal differentiations are eliminated, has led to the concept of impulse observer. We have shown that such an observer can be used in a feedback loop in the same way that a standard observer is used in state-space theory. We believe that many other familiar results may be generalized in this way with novel interpretations occurring in many instances.

REFERENCES

[1] F.R. Gantmacher, The Theory of Matrices, vol. 2, Chelsea, New York, 1964.

[2] S.L. Campbell, Singular Systems of Differential Equations, Pitman, London, 1980.

[3] H.H. Rosenbrock, "Structural properties of linear dynamical systems", Int. J. Control, 20 (1974), pp. 191-202.

[4] G.C. Verghese, B.C. Levy, T. Kailath, "A generalized state-space for singular systems", IEEE Trans. Automatic Control, 26 (1981), pp. 811-831.

[5] E.L. Yip, R.F. Sincovec, "Solvability, controllability, and observability of continuous descriptor systems", IEEE Trans. Automatic Control, 26 (1981), pp. 702-707.

[6] D. Cobb, "Feedback and pole-placement in descriptor-variable systems", Int. J. Control, 33 (1981), pp. 1135-1146.

STRONG CONTROLLABILITY AND STRONG OBSERVABILITY OF GENERALIZED DYNAMICAL SYSTEMS

GONGTIAN YAN
Beijing Institute
of Control Engineering
Beijing
People's Republic of China

T. J. TARN
Dept. of Systems Science
and Mathematics
Washington University
St. Louis, Missouri 63130

ABSTRACT

Strong controllability and strong observability of generalized state space systems are studied. Necessary and sufficient conditions for strong controllability, strong observability and strong irreducibility are given.

1. INTRODUCTION

In recent years, there has been a growing interest in the system theoretic problems of investigating generalized dynamical systems first considered by Rosenbrock [13] and then by Luenberger [14]. A number of papers, for example [2-5], have extended some results on regular polynomial systems developed by Rosenbrock [1] to generalized dynamical systems. These papers obtained results on infinite pole-zero structure, strong controllability and strong observability, strong equivalence, irreducible realizations and structural properties of system matrix and transfer function matrix. An alternative method which may be used to study generalized dynamic systems is to use the generalized inverse theory, particularly the so-called Drazin Inverse, to obtain explicit solutions [9].

Recently, the geometric method developed by Wonham [15] has also been used in the study of generalized dynamical systems from the state space theory point of view [10]. The major difference between generalized state space systems and regular state space system is that in the case of generalized state space systems the concepts of infinite frequencies and impulsive modes appear.

Kronecker's canonical form and minimal bases theory provide the detailed structural information of generalized state space systems. Computational methods for Kronecker's canonical form are available in [6, 12].

In this paper, we study strong controllability and strong observability of generalized dynamical systems. Specifically the Kronecker's theory on matrix pencils [11] and the minimal bases of rational vector spaces [6] are used to investigate the relationships between strong controllability and minimal indices of the system matrix. By duality the corresponding results on strong observability are also obtained. Combining these studies necessary and sufficient conditions for strong irreducibility are given which improves a necessary condition presented before by Verghese, Van Dooren and Kailath [2].

2. PRELIMINARIES

We consider the time-invariant, generalized state space system of the following form:

$$E\dot{x}(t) = Ax(t) + Bu(t)$$

$$y(t) = Cx(t) \tag{1}$$

This research was supported in part by the National Science Foundation under Grant Nos. ECS-8017184 and INT-8201554.

where x is an r-vector, u is an m-vector, y is a p-vector and E, A, B, C are real matrices of appropriate dimension. B is assumed to be of full column rank and C is assumed to be of full row rank. E is singular and $\det(sE-A) \not\equiv 0$.

The generalized order of the generalized state space system (1), denoted by f, is defined as $f \overset{\Delta}{=} \text{rank } E$ [2]. It is known that f is equal to the number of system modes (exponential and impulsive modes). The dimension of the kernel of E is the number of algebraic constraints of system (1). From matrix pencil theory, a regular pencil (i.e., $\det(sE - A) \not\equiv 0$) has the following canonical form, i.e., there exist two nonsingular constant matrices M and N such that

$$M(sE-A)N = \left[\begin{array}{c|ccc} sI_n-J & & 0 & \\ \hline & I_{n_1}-sN_1 & & \\ 0 & & \ddots & \\ & & & I_{n_k}-sN_k \end{array}\right], \tag{2}$$

where J is an nxn matrix and has Jordan form, $N_i (i=1,\ldots,k)$ is nilpotent and has the form

$$N_i = \begin{bmatrix} 0 & 1 & & 0 \\ & 0 & \ddots & \\ & & \ddots & 1 \\ 0 & & & 0 \end{bmatrix}_{n_i x n_i}$$

and of course $r = n + \sum_{i=1}^{k} n_i$.

In (2), $sI_n - J$ is the regular part and is corresponding to finite zeros of (sE − A). $In_i - sN_i$ is the singular part and is corresponding to infinite zeros of (sE − A). A free subsystem corresponding to $In_i - sN_i$ has $(n_i - 1)$ impulsive modes. We denote the total number of impulsive modes of (1) by $f_s = \sum_{i=1}^{k} (n_i - 1)$ and call f_s the singular order of (1). Note that

$$\text{generalized order}(f)=\text{regular order}(n)+\text{singular order}(f_s).$$

The index of nilpotency [11] of (1) denoted by $\bar{n}$, is $\bar{n} = \max_{i\varepsilon\{1,\ldots,k\}} \{n_i\}$. When $\bar{n} = 1$, there is no impulsive mode and it amounts to put $k = r - n$ algebraic constraints on (1). In such a case we have

$$\text{Rank } E = f = n = \delta|sE - A|, \tag{3}$$

where δ represents the degree of the characteristic polynomial of system (1). (3) is the condition of system (1) without impulsive modes.

The transfer function matrix, in general, consists of two parts:
(i) strictly proper rational matrix corresponding to the regular part, and
(ii) polynomial matrix corresponding to the singular part. From (2) we have

$$G(s) = CN[M(sE - A)N]^{-1}MB = \bar{C}(sI_{n} - J)^{-1}\bar{B} + \sum_{i=1}^{k} C_i(I_{n_i} - sN_i)^{-1}B_i, \quad (4)$$

where $CN = [\bar{C} \;\; C_1 \; .... \; C_k]$, $MB = [\bar{B}' \;\; B_1' \; .... \; B_k']'$. Since $(I_{n_i} - sN_i)^{-1} = I + sN_i + ... + s^{(n_i-1)} N_i^{(n_i-1)}$, it is clear that the degree of polynomial part is equal to or less than $(\bar{n} - 1)$. If for any i, $C_i N_i^{(n_i-1)} B_i = 0$ then (1) is uncontrollable or unobservable at infinity. Particularly, if the degree of the polynomial part is strictly less than $(\bar{n} - 1)$, then (1) is not strongly irreducible.

3. POLE AND ZERO STRUCTURE

The important properties of generalized state space systems distinguished from the properties of regular state space systems are the infinite frequencies and the corresponding impulsive modes. To understand these properties, we need to investigate the pole-zero structure of the system (1) at infinity. There exist at least three different but equivalent definitions of infinite poles and infinite zeros [1, 5, 7]. We adopt the one used by Rosenbrock [1]: the pole and zero structure of a rational matrix G(s) at infinity is defined as that of $G(\frac{1}{\lambda})$ at $\lambda = 0$.

For system (1), let

$$P_I(s) = [sE - A, -B],$$

$$P_0(s) = \begin{bmatrix} sE - A \\ C \end{bmatrix}.$$

The infinite input decoupling zeros and the infinite output decoupling zeros are the zeros of $P_I(\frac{1}{\lambda})$ and $P_0(\frac{1}{\lambda})$ at the origin respectively. The results on finite decoupling zeros can be extended to infinite decoupling zeros [8].

Note that for singular pencils, the Kronecker canonical form in general consists of four parts: [11]

$$\left[\begin{array}{c|c|ccc|ccc} sI - J & & & & & & 0 & \\ \hline & I - sN & & & & & & \\ \hline & & K_1(s) & & & & & \\ & & & \ddots & & & & \\ & & & & K_\alpha(s) & & & \\ \hline & 0 & & & & L_1(s) & & \\ & & & & & & \ddots & \\ & & & & & & & L_\beta(s) \end{array}\right] \quad (5)$$

where $$N = \begin{bmatrix} N_1 & & \\ & \ddots & \\ & & N_k \end{bmatrix},$$

$$K_i(s) = \begin{bmatrix} s & -1 & & 0 \\ & \ddots & \ddots & \\ 0 & & s & -1 \end{bmatrix}_{\varepsilon_i \times (\varepsilon_i + 1)}$$ - Kronecker column block,

$$L_j(s) = \begin{bmatrix} s & 0 & & \\ -1 & \ddots & \ddots & \\ & \ddots & \ddots & s \\ 0 & & & -1 \end{bmatrix}_{(\mu_j+1)\times\mu_j} \quad \text{- Kronecker row block.}$$

The first block is the Jordan block and corresponds to the finite elementary divisors, the second block is the nilpotent block and corresponds to infinite elementary divisors, the Kronecker column block corresponds to minimal column indices, and the Kronecker row block corresponds to minimal row indices. We summarize the pole and zero structure of each type of blocks as follows: (i) In the first block, number of finite zeros = number of infinite poles = dimension of the block. (ii) Concerning the second block, there are only $\sum_{i=1}^{k} (n_i - 1)$ infinite poles and $\sum_{i=1}^{k} (n_i - 1)$ infinite zeros where n_i is the dimension of N_i. (iii) $K_i(s)$ has only ε_i infinite poles, (iv) $L_j(s)$ has only μ_j infinite poles.

Given a rational matrix R(S), let $N_p(R)$ denotes the total number of poles (finite and infinite) of R(S), $N_z(R)$ denotes the total number of zeros (finite and infinite) of R(S) and $\gamma(R)$ denotes the sum of minimal column indices and minimal row indices of R(S), then [2, Theorem 3].

$$N_p(R) - N_z(R) = \gamma(R). \tag{6}$$

4. CONTROLLABILITY AND OBSERVABILITY

In this section we study strong controllability and strong observability of generalized state space systems. Specifically the Kronecker's theory on matrix pencils [11] and the minimal bases of rational vector spaces [6] are used to investigate the relationships between strong controllability and minimal indices of $P_I(s)$. Using the concept of duality, the corresponding results on strong observability is also obtained. Combining these studies necessary and sufficient conditions for strong irreducibility are given which improves a necessary condition presented before by Verghese, Van Dooren and Kailath [2].

Definition 1. [4] (i) An impulsive mode is said to be controllable if it can be excited from zero initial conditions using nonimpulsive inputs. (ii) The system whose impulsive modes are controllable is said to be controllable at infinity. (iii) The system is said to be strongly controllable, if it is controllable in the regular sense and controllable at infinity.

Definition 2. [4] (i) The impulsive mode is said to be unobservable if the output corresponding to this impulsive free response mode is zero. (ii) The system without unobservable impulsive modes is said to be observable at infinity. (iii) The system is said to be strongly observable if it is observable in the regular sense and observable at infinity.

Remark 1. It is evident that: (i) System (1) is strongly controllable iff $P_I(s)$ has no finite and infinite zeros. (ii) System (1) is strongly observable iff $P_0(s)$ has no finite and infinite zeros. [5]

It is well known that the pole and zero structure is invariant under strong equivalence and that the strong controllability and strong observability are intimately connected with the pole and zero structure. In the following theorems we present some results on strong controllability and strong observability based on canonical forms of $P_I(s)$ and $P_0(s)$ respectively.

Theorem 1. System (1) is strongly controllable iff the Kronecker canonical form of $P_I(s)$ has the following properties: (i) It has no finite elementary divisors. (ii) Index of nilpotency of the nilpotent block is not more than one.

Proof. Sufficiency. If the canonical form of $P_I(s)$ has properties (i) and (ii), then $P_I(s)$ has no finite and infinite zeros. Thus $P_I(s)$ is strongly controllable.

Necessity. Observe that the assumption system (1) satisfies $\det(sE - A) \not\equiv 0$ implies the canonical form of $P_I(s)$ does not have minimal row indices. To see this let us suppose that there exist minimal row indices. Then this implies $[sE - A, - B]$ is row dependent. So is $(sE - A)$. Thus $\det|sE - A| \equiv 0$. Henceforth we only consider those $P_I(s)$ whose canonical form does not have minimal row indices. (i) If there exists finite elementary divisors, then $P_I(s)$ has finite input decoupling zeros. This implies that the system is not strongly controllable. (ii) Suppose one of the nilpotent block has dimension larger than one. This implies that $P_I(s)$ has infinite input decoupling zeros. Thus the system is not strongly controllable.

Theorem 2. System (1) is strongly observable iff the Kronecker canonical form of $P_0(s)$ has the following properties: (i) It has no finite elementary divisors, (ii) Index of nilpotency of the nilpotent block is not more than one.

Proof. By duality.

Remark 2. It is easy to show that system (1) is strongly controllable iff the sum of the minimal column indices of $P_I(s)$ equals the number of infinite poles of $P_I(s)$ and system (1) is strongly observable iff the sum of the minimal row indices of $P_0(s)$ equals the number of infinite poles of $P_0(s)$.

Theorem 3. System (1) is strongly controllable ($P_I(s)$ is strongly irreducible) iff $\sum_{i=1}^{m} \varepsilon_i = f$, where ε_i is the minimal column indices of $P_I(s)$ and f is the generalized order of system (1).

Proof. Necessity. From Theorem 1 we know that $P_I(s)$ has the following canonical form.

$$M[sE-A,-B]N = \begin{array}{l} k\left\{\vphantom{\begin{matrix}1\\ \ddots\\ 1\end{matrix}}\right. \\ \vphantom{\begin{matrix}K_1(s)\\ \ddots\\ K_m(s)\end{matrix}} \end{array} \!\!\left[\begin{array}{c|c} \begin{matrix}1 & & \\ & \ddots & \\ & & 1\end{matrix} & 0 \\ \hline 0 & \begin{matrix}K_1(s) & & \\ & \ddots & \\ & & K_m(s)\end{matrix} \end{array}\right],$$

$$K_i(s) = \begin{bmatrix} s & 1 & & 0 \\ & s & 1 & \\ & & \ddots & \ddots \\ 0 & & s & 1 \end{bmatrix}_{\varepsilon_i \times (\varepsilon_i + 1)} . \tag{7}$$

So we have $\sum_{i=1}^{m} \varepsilon_i = r - k$. Equation (7) can be rewritten as

$$sM[E \quad 0]N + M[-A \quad -B]N$$

$$= s\begin{bmatrix} \begin{matrix} 0 & & \\ & \ddots & \\ & & 0 \end{matrix} & & & \\ & \begin{matrix} 1 & 0 & & \\ & \ddots & \ddots & \\ & & 1 & 0 \end{matrix} & & \\ & & \ddots & \\ & & & \begin{matrix} 1 & 0 & & \\ & \ddots & \ddots & \\ & & 1 & 0 \end{matrix} \end{bmatrix} + \begin{bmatrix} \begin{matrix} 1 & & \\ & \ddots & \\ & & 1 \end{matrix} & & & \\ & \begin{matrix} 0 & 1 & & \\ & \ddots & \ddots & \\ & & 0 & 1 \end{matrix} & & \\ & & \ddots & \\ & & & \begin{matrix} 0 & 1 & & \\ & \ddots & \ddots & \\ & & 0 & 1 \end{matrix} \end{bmatrix} \tag{8}$$

From (8) it is clear that the rank of the coefficient matrix of s equals the rank of E(=f). This implies that $\sum_{i=1}^{m} \varepsilon_i = r - k = f$.

Sufficiency. If $f = \sum_{i=1}^{m} \varepsilon_i$, then equation (7) will be the canonical form of $P_I(s)$. (Otherwise $f > \sum_{i=1}^{m} \varepsilon_i$). Thus $P_I(s)$ has no finite and infinite zeros. This implies that $P_I(s)$ is strongly controllable.

Theorem 4. System (1) is strongly observable ($P_0(s)$ is strongly irreducible) iff $\sum_{j=1}^{p} \mu_j = f$, where μ_j is the minimal row indices of $P_0(s)$.

Proof. By duality.

Now we are ready to present the main theorems of this paper.

Theorem 5. The system matrix P(s) of (1) is strongly irreducible (i.e., system (1) is strongly controllable and strongly observable) iff the sum of the minimal column indices (ΣMCI) of $P_I(s)$ = the sum of the minimal row indices (ΣMRI) of $P_0(s)$ = f.

Proof: The theorem follows immediately from Theorems 3 and 4.

Theorem 6. The system matrix P(s) of (1) is strongly irreducible iff ΣMCI of G(s) + ΣMRI of G(s) = f − $N_z(G)$ where G(s) is the transfer function matrix of (1).

Proof. Necessity. From (6) we have

$$\Sigma\text{MCI of } G(s) + \Sigma\text{MRI of } G(s) = N_p(G) - N_z(G).$$

P(s) is strongly irreducible implies that $f = N_p(G)$.

Sufficiency. Assume that the generalized order of (1) is f with system matrix P(s) and transfer function matrix G(s). By assumption we have

$$\Sigma\text{MCI of } G(s) + \Sigma\text{MRI of } G(s) = f - N_z(G).$$

We want to show that P(s) is strongly irreducible. Suppose not then (1) is either not strongly controllable or not strongly observable or both. In any case by cancelling all decoupling zeros of P(s) we can find a system matrix $\tilde{P}(s)$ which has the same transfer function matrix G(s) [1, 5]. Moreover the generalized order f' of the corresponding generalized state space system of $\tilde{P}(s)$ is less than f and $\tilde{P}(s)$ is strongly irreduciable. Thus we have

$$\Sigma\text{MCI of } G(s) + \Sigma\text{MRI of } G(s) = f' - N_z(G) < f - N_z(G).$$

contradiction.

Verghese, Van Dooren and Kaileth [2] made use of compression transformation and obtained a criterion for controllability at infinity as follows: Using row compression one get

$$M[sE-A,\ -B] = \begin{bmatrix} sE_1 - A_1 & -B_1 \\ -A_2 & -B_2 \end{bmatrix}$$

where E_1 is full row rank. System (1) is controllable at infinity if $\begin{bmatrix} E_1 & 0 \\ -A_2 & -B_2 \end{bmatrix}$ has full row rank. We extend this result below.

Theorem 7. The following statements are equivalent: (i) System (1) is controllable at infinity. (ii) In the canonical form of $P_I(s)$, the nilpotent block has index of nilpotency equals one. (iii) $P_I(s)$ is row reduced. (iv) $b_1, \ldots, b_k$ are linearly independent, where b_i's are the last rows of B_i's and B_i's are from the following canonical form of $P_I(s)$.

$$\left[\begin{array}{c|c|c} sI_n - J & & B_0 \\ \hline & \begin{matrix} I_{n_1} - sN_1 & & \\ & \ddots & \\ & & I_{n_k} - sN_k \end{matrix} & \begin{matrix} B_1 \\ \vdots \\ B_k \end{matrix} \end{array}\right] . \qquad (9)$$

Proof. (i) => (ii) is trivial. (ii) => (iii). From assumption (ii), the Kronecker canonical form of $P_I(s)$ is

$$\left[\begin{array}{c|c|c} sI_n - J & & \\ \hline & \begin{matrix} 1 & & \\ & \ddots & \\ & & 1 \end{matrix} & \\ \hline & & \begin{matrix} k_1(s) & & \\ & \ddots & \\ & & K_m(s) \end{matrix} \end{array}\right] \quad \text{where } K_i(s) = \begin{bmatrix} s & 1 & & \\ & \ddots & \ddots & \\ & & \ddots & \ddots \\ & & s & 1 \end{bmatrix} . \qquad (10)$$

Examining (10) it is clear that $P_I(s)$ is row-reduced.

(iii) => (iv). Using strong equivalence transformation we bring $P_I(s)$ to the form (9). Since row reducedness is invariant under strong equivalence transformation, hence (9) is row reduced. This requires that the leading row coefficient matrix of (9) has full row rank, thus $b_1, \ldots, b_k$ are linearly independent.

(iv) => (i). We want to show that $P_I(s)$ has no infinite zeros. Since all infinite zeros are contained in the singular part of (9), we need only to consider the singular part of (9). The infinite zeros of the singular part of (9) equals the zeros at the origin of the numerator matrix when we set $s = \frac{1}{\lambda}$ in the singular part of (9). Let $\lambda = 0$, the numerator matrix has full row rank. This implies that the numerator matrix has no zeros at the origin and hence $P_I(s)$ has no infinite zeros.

Remark 3. By duality we can obtain the corresponding results on observability at infinity.

For regular systems, Rosenbrock [1] proved that $[sI_n-A, B]$ is controllable iff there does not exist a non-zero vector v such that

$$v^*[s_iI_n-A, B] = 0,$$

where s_i, $i = 1, \ldots, n$ are eigenvalues of A. We extend this result to generalized state space systems.

Theorem 8. System (1) is strongly controllable iff there does not exist a non-zero vector v such that

$$v^*[s_iE-A, -B] = 0$$

where s_i, $i = 1, \ldots, f$ are finite or infinite eigenvalues of (E, A). In other words there are no left eigenvectors of (E, A) which is perpendicular to the image space of B.

Proof. For finite eigenvalues simply applying Rosenbrock's result [1, Theorem 6.2] on regular systems.

For infinite eigenvalues, the corresponding left eigenvectors of (E,A) are in the left null space of E [16]. Using row compression on [E 0] - [A B] we get

$$\begin{bmatrix} E_1 & 0 \\ 0 & 0 \end{bmatrix} - \begin{bmatrix} A_1 & B_1 \\ A_2 & B_2 \end{bmatrix},$$

where E_1 has full row rank. Therefore the left eigenvectors must have the form $v = [\underbrace{0 \ldots 0}_{f} \ x \ldots x]^T$. From [2] we know that system (1) is controllable at infinity iff $\begin{bmatrix} E_1 & 0 \\ A_2 & B_2 \end{bmatrix}$ has full row rank. This is equivalent to the requirement that $v = [0, \ldots, 0]^T$.

Remark 4. The corresponding result on strong observability can be obtained via the dual concept.

Acknowledgement. The authors would like to thank Mr. Wijesooriya Dayawansa for his helpful suggestions.

REFERENCES

[1] H. H. Rosenbrock, "State-space and Multivariable Theory", 1970.

[2] G. Verghese, P. Van Dooren and T. Kailath, "Properties of the System Matrix of a Generalized State-space System", Int. J. Control, Vol. 30, p. 238-243, 1979.

[3] G. Verghese and T. Kailath, "Generalized Dynamic Systems", 18th IEEE Conf. on Decision and Control, December 1979.

[4] G. Verghese, B. C. Levy and T. Kailath, "A Generalized State-space for Singular Systems", IEEE Trans. on Automatic Control, AC-26, No. 4, August 1981.

[5] G. Verghese, "Infinite-frequency Behavior in Generalized Dynamical Systems", Ph.D. dissertation, Dept. of Electrical Eng., Stanford Univ., December 1978.

[6] G. D. Forney, "Minimal Bases of Rational Vector-space with Applications to Multivariable Linear Systems", SIAM J. Control, Vol. 13, No. 3, May 1975.

[7] A. C. Pugh and P. A. Ratcliffe, "Infinite Frequency Interpretations of Minimal Bases", Int. J. Control, Vol. 32, No. 4, p. 581-588, 1980.

[8] P. M. G. Ferreira, "Infinite System Zeros", Int. J. Control, Vol. 32, No. 4, p. 731-735, 1980.

[9] S. L. Campbell, C. D. Mayer and N. J. Rose, "Applications of the Inverse to Linear Systems of Differential Equations with Singular Constant Coefficients", SIAM J. Appl. Math., Vol. 31, No. 3, p. 411-425, November 1976.

[10] D. Cobb, "Feedback and Pole Placement in Description Variable Systems", Int. J. Control, Vol. 33, No. 6, p. 1135-1146, 1981.

[11] Gantmacher, "Matrix Theory", Vol. 2, 1959.

[12] P. Van Dooren, "Computation of Kronecker's Canonical Form of a Singular Pencil", Linear Algebra and Its Applications, 27, p. 103-140, 1979.

[13] H. H. Rosenbrock, "Structural Properties of Linear Dynamic Systems", Int. J. Control, Vol. 20, No. 2, p. 191-202, 1974.

[14] D. G. Luenberger, "Dynamical Equations in Descript Form", IEEE Trans. on Automatic Control, Vol. 22, No. 3, June 1977.

[15] W. M. Wonham, "Linear Multivariable Control: A Geometric Approach", Springer-Verlag, 1979.

[16] G. Verghese and T. Kailath, "Eigenvector Chains for Finite and Infinite Zeros of Rational Matrices", Proceedings of the 18th IEEE Conference on Decision and Control, Fort Lauderdale, FL, Dec. 1979.

INFINITE-TIME REGULATORS FOR SINGULARLY PERTURBED DIFFERENCE EQUATIONS[1]

B. LITKOUHI, AND H.K. KHALIL
Michigan State University, East Lansing, MI 48824

SUMMARY

In a recent paper [1], Blankenship studied linear quadratic optimal control problems for singularly perturbed difference equations when the cost function is defined on a finite-time period. He, essentially, showed that the basic features of the singular perturbation approach to continuous-time problems can be extended to discrete-time ones. Our paper will extend on Blankenship's work. In this paper we consider the infinite-time optimal control problem and study the asymptotic behavior of the resulting algebraic Riccati equations as the perturbation parameter tends to zero.

It is shown that the asymptotic behavior of the infinite-time problem does not follow as a limiting case of the finite-time problem. A special scaling of the solution of the Riccati equation is employed and is shown to be appropriate to expand solution as a series in the perturbation parameter.

It is also shown that a near-optimal solution can be obtained by applying slow-fast decompositions as in the continuous-time work of Chow and Kokotovic [2]. In our case the slow optimal control problem will be a continuous-time problem, while the fast optimal control problem will be a discrete-time problem.

I. Problem Statement

Consider the linear time-invariant discrete-time system

$$x(n+1) = (I+\varepsilon A)x(n) + \varepsilon Bz(n) + \varepsilon Fu(n) \tag{1a}$$

$$z(n+1) = Cx(n) + Sz(n) + Gu(n) \tag{1b}$$

$$y(n) = D_1x(n) + D_2z(n) + Mu(n) \tag{1c}$$

where $\varepsilon>0$ is a small positive parameter, the state vector comprises the m_1 and m_2 dimensional vectors x and z, the control u is an r-dimensional vector and the output y is a k vector. The initial states x(0) and z(0) are given. The controls u(n) are to be selected to minimize the performance index

$$J = \varepsilon \sum_{n=0}^{\infty} [y^T(n)y(n) + u^T(n)R\,u(n)], \qquad R = R^T>0. \tag{2}$$

For simplicity the matrices A, B, C, F, G, S, D_1, D_2, M and R are taken to be independent of ε but they could be analytic functions of ε and the problem would be treated in the same way. The finite-time version of this problem was considered by Blankenship [1] who investigated the asymptotic behavior of the optimal solution as $\varepsilon \to 0$. The asymptotic behavior of the infinite time problem we are interested in here does not follow from Blankenship's study as a limiting case when the terminal time N tends to ∞. To see this one should observe that eqn. (2.34) of [1] gives an asymptotic formula for the solution of the associated Riccati equation which is

---

[1] This work was supported in part by the U.S. Department of Energy, Electric Energy Systems Division, under contract no. DE-AC01-80RA50425.

proportional to N; so it blows up as N $\longrightarrow\infty$. By appropriately scaling the solution of the Riccati equation in the next section we will be able to overcome this difficulty.

The infinite-time regulator problem (1), (2) could be a result of discretization or Sampled-data control of infinite-time regulators for singularly perturbed continuous time systems, using the methods of [3]. The details are similar to the finite-time example presented by Blankenship [1]. The form of the performance index is a little bit more complicated than the one studied by Blankenship because of the presence of the matrix M. When M=0, the performance index J reduces to the one studied by Blankenship. The current form is chosen to accommodate the sampled-data control case where J is obtained by discretizing an integral performance index of a continuous-time problem. It was shown in [3] that discretizing an integral performance index with quadratic terms in the state and control (and no cross product terms) results in a discrete-time performance index of the chosen form with $M \neq 0$.

II. Asymptotic Behavior of the Optimal Solution

The optimal control of the system (1) with performance index (2) is given by [3]

$$u(n) = -[R+M^TM+\mathcal{B}^TP\mathcal{B}]^{-1}[\mathcal{B}^TP\mathcal{A}+M^TD] \begin{bmatrix} x(n) \\ z(n) \end{bmatrix} \tag{3}$$

where P is a stabilizing solution of the discrete-time algebraic Riccati equation

$$P = D^TD+\mathcal{A}^TP\mathcal{A}-[\mathcal{A}^TP\mathcal{B}+D^TM][R+M^TM+\mathcal{B}^TP\mathcal{B}]^{-1}[\mathcal{B}^TP\mathcal{A}+M^TD] \tag{4}$$

and where

$$\mathcal{A} = \begin{bmatrix} I+A & \varepsilon B \\ C & S \end{bmatrix}, \quad \mathcal{B} = \begin{bmatrix} \varepsilon F \\ G \end{bmatrix}, \quad D = [D_1, D_2].$$

In studying the asymptotic behavior of the Riccati equation (4), we seek the matrix P in the form

$$P = \begin{bmatrix} P_1/\varepsilon & P_2 \\ P_2^T & P_3 \end{bmatrix}. \tag{5}$$

The form (5) plays a crucial role in studying the solution of the Riccati equation (4). It is different from the form used by Blankenship since he used

$$P = \begin{pmatrix} P_1 & P_2 \\ P_2^T & P_3 \end{pmatrix} \tag{6}$$

Substituting (5) into (4) and partitioning the Riccati equation yields

$$0 = f_1(P_1, P_2, P_3, \varepsilon) \tag{7}$$

$$P_2 = f_2(P_1, P_2, P_3, \varepsilon) \tag{8}$$

$$P_3 = f_3(P_1,P_2,P_3,\varepsilon) \tag{9}$$

where the functions f_1, f_2 and f_3 are defined in Appendix A. To study the solution of (7)-(9) near $\varepsilon=0$, it is natural to start by setting $\varepsilon=0$ in (7)-(9). This yields

$$\begin{aligned} 0 = {} & P_1(0)A+P_2(0)C+A^TP_1(0)+C^TP_2{}^T(0)+C^TP_3(0)C+D_1{}^TD_1 \\ & - [P_1(0)F+P_2(0)G+C^TP_3(0)G+D_1{}^TM][R+M^TM+G^TP_3(0)G]^{-1} \\ & \times [F^TP_1(0)+G^TP_2{}^T(0)+G^TP_3(0)C+M^TD_1], \end{aligned} \tag{10}$$

$$\begin{aligned} P_2(0) = {} & P_1(0)B+P_2(0)S+C^TP_3(0)S+D_1{}^TD_2 \\ & -[P_1(0)F+P_2(0)G+C^TP_3(0)G+D_1{}^TM][R+M^TM+G^TP_3(0)G]^{-1} \\ & [G^TP_3(0)S+M^TD_2], \end{aligned} \tag{11}$$

$$\begin{aligned} P_3(0) = {} & S^TP_3(0)S+D_2{}^TD_2-[S^TP_3(0)G+D_2{}^TM][R+M^TM+G^TP_3(0)G]^{-1} \\ & [G^TP_3(0)S+M^TD_2]. \end{aligned} \tag{12}$$

Equation (12) is a discrete-time algebraic Riccati equation. It is well-known [4] that if the pair $[S-G(R+M^TM)^{-1}M^TD_2,G]$ is stabilizable and the pair $[S-G(R+M^TM)^{-1}M^TD_2,\sqrt{D_2{}^TD_2-D_2{}^TM(R+M^TM)^{-1}M^TD_2}]$ is detectable then (12) has a unique positive semidefinite solution. It is obvious that the stabilizability of $[S-G(R+M^TM)^{-1}M^TD_2,G]$ is equivalent to the stabilizability of $[S,G]$. Moreover, using the matrix identity $I-M(R+M^TM)^{-1}M^T=(I+MR^{-1}M^T)^{-1}$ it can be shown that the detectability of

$\left[S-G(R+M^TM)^{-1}M^TD_2,\sqrt{D_2{}^TD_2-D_2{}^TM(R+M^TM)^{-1}M^TD_2}\right]$ is equivalent to the detectability of $[S-G(R+M^TM)^{-1}M^TD_2,\ D_2]$ which is equivalent to the detectability of $[S,D_2]$. Thus we assume that the triple $[S,G,D_2]$ is stabilizable-detectable which guarantees the existence of $P_3(0)\geq 0$. Furthermore, from the properties of Riccati equations [4] we have the stability property

$$|\lambda(\alpha_3)|<1 \tag{13}$$

where $\alpha_3 = S-G[R+M^TM+G^TP_3(0)G]^{-1}[G^TP_3(0)S+M^TD_2]$.

We turn now to equation (11) and notice that $P_2(0)$ can be expressed in terms of $P_1(0)$ and $P_3(0)$ as

$$P_2(0) = L_1+P_1(0)L_2 \tag{14}$$

where

$$L_1 = \left\{ D_1{}^T D_2 + C^T P_3(0)S - [C^T P_3(0)G + D_1{}^T M][R + M^T M + G^T P_3(0)G]^{-1} [G^T P_3(0)S + M^T D_2] \right\} L_3{}^{-1}$$

$$L_2 = \left\{ B - F[R + M^T M + G^T P_3(0)G]^{-1} [G^T P_3(0)S + M^T D_2] \right\} L_3{}^{-1}$$

$$L_3 = I - \alpha_3$$

and where the nonsingularity of L_3 follows from the stability property (13). Substituting (14) into (10) yields

$$0 = P_1(0)\hat{A} + \hat{A}^T P_1(0) + \hat{Q} - P_1(0)\hat{B}\hat{R}^{-1}\hat{B}^T P_1(0) \tag{15}$$

where

$$\hat{B} = F + L_2 G,$$

$$\hat{R} = R + M^T M + G^T P_3(0)G,$$

$$\hat{A} = A + L_2 C - \hat{B}\hat{R}^{-1}[G^T L_1{}^T + G^T P_3(0)C + M^T D_1],$$

and

$$\hat{Q} = D_1{}^T D_1 + L_1 C + C^T L_1{}^T + C^T P_3(0)C - [L_1 G + C^T P_3(0)G + D_1{}^T M]\hat{R}^{-1}[G^T L_1{}^T + G^T P_3(0)C + M^T D_1].$$

Equation (15) is a continuous-time algebraic Riccati equation. We assume that the triple $(\hat{A}, \hat{B}, \sqrt{\hat{Q}})$ is stabilizable-detectable. This guarantees [4] that (15) has a unique positive semidefinite solution $P_1(0) \geq 0$ and

$$\mathrm{Re}\{\lambda(\alpha_1)\} < 0 \tag{16}$$

where $\alpha_1 = \hat{A} - \hat{B}\hat{R}^{-1}\hat{B}^T P_1(0)$.

Thus we have established the existence of the solution of (7)-(9) at $\varepsilon=0$. For ε near zero, the existence of the solution is established by applying the Implicit Function Theorem where the nonsingularity of $(I-\alpha_3)$ and α_1, which follow from the stability properties (13) and (16) respectively, are used to show that the Ferchet derivatives of $f_1, f_2 - P_2$ and $f_3 - P_3$ with respect to P_1, P_2 and P_3 at $\varepsilon=0$ are invertible. By the Implicit Function Theorem it follows that there exists $\varepsilon^* > 0$ such that for all $\varepsilon \epsilon (0, \varepsilon^*)$, $P_1(\varepsilon)$, $P_2(\varepsilon)$ and $P_3(\varepsilon)$ exist and are analytic in ε at $\varepsilon=0$. It remains now to show that this solution is stabilizing. This, however, follows immediately by applying the stability criterion derived in Appendix B to the closed-loop system where the stability properties (13) and (16) are used. Our conclusions are summarized in the following theorem:

Theorem 1: Assume that

Condition a: The triple $(\hat{A},\hat{B},\sqrt{\hat{Q}})$ is stabilizable-detectable in the continuous-time sense; i.e., every eigenvalue of $\hat{A}$ which lies in the closed right-half complex plane is controllable and observable,

Condition b: The Triple (S,G,D_2) is stabilizable-detectable in the discrete-time sense; i.e., every eigenvalue of S which have modulus greater or equal to one is controllable and observable.

Then there exists $\varepsilon^* > 0$ such that for all $0<\varepsilon<\varepsilon^*$, the Riccati equation (4) has a unique positive semidefinite stabilizing solution. Furthermore, the solution has a power series expansion at $\varepsilon=0$, that is

$$P = \sum_{i=0}^{\infty} \frac{\varepsilon^i}{i!} \begin{bmatrix} P_1^{(i)}/\varepsilon & P_2^{(i)} \\ P_2^{(i)T} & P_3^{(i)} \end{bmatrix}.$$

One unpleasant feature of Theorem 1 is that Condition a is dependent on $P_3(0)$, the solution of the discrete-time Riccati equation (12). It will be shown later that the matrices $\hat{A},\hat{B}$ and $\hat{Q}$ are indeed independent of $P_3(0)$. For the time-being, however, let us assume that Conditions a and b hold so that Theorem 1 provides us with a reasonable way to approximate the optimal control for small ε. An approximate state feedback control is defined by

$$u(n) = -[R+M^TM+\mathscr{B}^TP'\mathscr{B}]^{-1}[\mathscr{B}^TP'\mathscr{A}+M^TD] \begin{bmatrix} x(n) \\ z(n) \end{bmatrix} \tag{17}$$

where $\overline{P}$ is obtained by truncating the expansion of P; i.e.,

$$P' = \sum_{i=0}^{N-1} \frac{\varepsilon^i}{i!} \begin{bmatrix} P_1^{(i)}/\varepsilon & P_2^{(i)} \\ P_2^{(i)T} & P_3^{(i)} \end{bmatrix} \tag{18}$$

The near optimality of the control low (17), (18) is established in the following theorem.

Theorem 2: Under conditions (a) and (b), the use of feedback control (17), (18) is $O(\varepsilon^{2N})$ near-optimal in the sense that

$$J-J_{opt} = O(\varepsilon^{2N}) \tag{19}$$

where J is the value of the performance criterion when (17), (18) is used, while J_{opt} is its optimal value.

Proof

Let us represent the optimal feedback control as

$$u^0_{opt} = -(F_1^0 \;\; F_2^0)X(n) \triangleq \mathscr{F}^0 X(n), X(n) = \begin{pmatrix} x(n) \\ z(n) \end{pmatrix}, \tag{20}$$

then

$$J_{opt} = \varepsilon X^T(0)PX(0) \tag{21}$$

where P satisfies equation (4). Similarly, the approximate feedback control is represented as

$$u = (F_1 \quad F_2)X(n) \triangleq -\mathcal{F}X(n). \tag{22}$$

It can be easily verified that

$$F_i^0 - F_i = 0(\varepsilon^N) \;,\; i = 1,2 \tag{23}$$

The closed-loop system under the feedback control (22) is

$$X(n+1) = (\mathcal{A}-\mathcal{B}\mathcal{F})X(n) \tag{24}$$

and the performance index is

$$J = \varepsilon X^T(0)P'X(0) \tag{25}$$

where P' satisfies the Lyapunov equation

$$P' = (\mathcal{A}-\mathcal{B}\mathcal{F})^T P'(\mathcal{A}-\mathcal{B}\mathcal{F}) + D^TD + \mathcal{F}^T(R+M^TM)\mathcal{F} + D^TM\mathcal{F} + \mathcal{F}^TM^TD \tag{26}$$

Subtracting (4) from (26) yields

$$P'-P = (\mathcal{A}-\mathcal{B}\mathcal{F})^T(P'-P)(\mathcal{A}-\mathcal{B}\mathcal{F}) + [\mathcal{F}-\mathcal{F}^0]^T(R+M^TM+\mathcal{B}^TP\mathcal{B})[\mathcal{F}-\mathcal{F}^0]. \tag{27}$$

But the term inside the least bracket is

$$\begin{pmatrix} F_1 - F_1^0 \\ \\ F_2 - F_2^0 \end{pmatrix} = 0(\varepsilon^N).$$

Letting $P'-P = V$, (27) becomes

$$V = (\mathcal{A}-\mathcal{B}\mathcal{F})^T V(\mathcal{A}-\mathcal{B}\mathcal{F}) + 0(\varepsilon^{2N}). \tag{28}$$

By application of the implicit function Theorem we can show that P' possesses a power series expansion at $\varepsilon=0$. So V can be expanded as follows:

$$V = \sum_{i=0}^{\infty} \frac{\varepsilon^i}{i!} \begin{bmatrix} V_1^{(i)}/\varepsilon & V_2^{(i)} \\ V_2^{(i)T} & V_3^{(i)} \end{bmatrix} \tag{29}$$

Partioning (28) and matching the zero order terms yields

$$0 = V_1^{(0)}(A-FR_0^{-1}N_0^T) + V_2^{(0)}W + (A-FR_0^{-1}N_0^T)^TV_1^{(0)}$$

$$+W^TV_2^{(0)T} + W^TV_3^{(0)}W \tag{30-a}$$

$$V_2^{(0)} = V_1^{(0)}J + V_2^{(0)}\alpha_3 + W^TV_3^{(0)}\alpha_3 \tag{30-b}$$

$$V_3^{(0)} = \alpha_3 V_3^{(0)} \alpha_3 \tag{30-c}$$

where

$$R_0 = R+M^TM+G^TP_3(0)G$$

$$D_0 = S^TP_3(0)G+D_2{}^TM$$

$$N_0 = P_1(0)F+P_2(0)G+C^TP_3(0)G+D_1{}^TM$$

$$J = B-FR_0{}^{-1}D_0{}^T$$

$$W = C-GR_0{}^{-1}N_0{}^T.$$

Stability of α_3 implies $V_3^{(0)} = 0$.

Evaluating $V_2^{(0)}$ from (30-b) and substituting into (30-a) and using (30-c) yields

$$V_1^{(0)}\alpha_1+\alpha_1^TV_1^{(0)} = 0$$

but α_1 is stable which implies $V_1^{(0)} = 0$.

Repeating the same argument, it can be shown that $V_i^{(2N-1)} = 0$, $i = 1,2,3$ which completes the proof of the theorem.

III. Slow-Fast Decompositions

System (1) is singular as a function of ε; i.e., we can observe order reduction and separation of time scales as $\varepsilon \to 0$. Blankenship [1] showed that for small ε the variables can be decomposed into slow and fast variables. Using Blankenship's time decompositions, slow and fast subproblems are defined in a way similar to that of Chow and Kokotovic [2].

Slow Subproblem: The slow variables, evolving in a slow time scale εn, satisfy the outer solution

$$X(\varepsilon n+\varepsilon)-X(\varepsilon n) = \varepsilon AX(\varepsilon n) + \varepsilon BZ(\varepsilon n)+ \varepsilon FU(\varepsilon n) \tag{31a}$$

$$Z(\varepsilon n) = CX(\varepsilon n) + SZ(\varepsilon n)+GU(\varepsilon n) \tag{31b}$$

$$Y(\varepsilon n) = D_1X(\varepsilon n)+D_2Z(\varepsilon n)+MU(\varepsilon n). \tag{31c}$$

Dividing (31a) by ε and letting $\varepsilon \to 0$ yields

$$\frac{d\overline{X}}{dt} = A\ \overline{x}(t)+B\ \overline{z}(t)+F\ \overline{u}(t) \tag{32a}$$

$$\overline{Z}(t) = C\ \overline{x}(t)+S\ \overline{z}(t)+G\ \overline{u}(t) \tag{32b}$$

$$\overline{Y}(t) = D_1\ \overline{x}(t)+D_2\ \overline{z}(t)+M\ \overline{u}(t) \tag{32c}$$

Assuming that (I-S) is nonsingular, (32b) can be used to eliminate $\overline{z}(t)$ from (32a) and (32c) resulting in the slow subsystem

$$\frac{dx_s}{dt} = A_s x_s(t)+B_s u_s(t) \tag{33a}$$

$$y_s(t) = C\ x\ (t)+D\ u\ (t) \tag{33b}$$

where $x_s = \overline{x}$, $y_s = \overline{y}$, $u_s = \overline{u}$ and $x_s(0) = x(0)$, and where

$$A_s = A+B(I-S)^{-1}C, \qquad B_s = F+B(I-S)^{-1}G,$$

$$C_s = D_1+D_2(I-S)^{-1}C, \quad D_s = M+D_2(I-S)^{-1}G.$$

We define a slow performance index J_s as

$$J_s = \int_0^\infty (y_s^T(t)y_s(t)+u_s^T(t)R\ u_s(t)dt \tag{34}$$

where in obtaining (34) we used the limiting relation

$$\lim_{\varepsilon \to 0} \ \varepsilon \sum_{n=0}^{\infty} \overline{U}^T(\varepsilon n)R\ \overline{U}(\varepsilon n) = \int_0^\infty \overline{u}^T(t)R\overline{u}(t)dt.$$

The slow problem defined by (33), (34) is a continuous-time regulator problem identical to the slow problem of Chow and Kokotovic [2]. Following [2] we assume that

Condition a′: the triple (A_s,B_s,C_s) is stabilizable-detectable in the continuous-time sense; i.e., every eigenvalue of A_s which lies in the closed right-half complex plane is controllable and observable. Under Condition a′, the optimal feedback control law of the slow problem (33), (34) is given by

$$u_s(t) = -R_s^{-1}[D_s^T C_s+B_s^T P_s]\ x_s(t) \triangleq -F_s x_s(t) \tag{35}$$

where $R_s = R+D_s^T D_s$ and P_s is the unique positive semidefinite stabilizing solution of the continuous-time algebraic Riccati equation

$$\begin{aligned} 0 = {} & P_s(A_s-B_sR_s^{-1}D_s^TC_s) + (A_s-B_sR_s^{-1}D_s^TC_s)^TP_s \\ & - P_sB_sR_s^{-1}B_s^TP_s + C_s^T(I-D_sR_s^{-1}D_s^T)C_s \end{aligned} \tag{36}$$

Fast Subproblem: Assume that the variables x(n), z(n), y(n) and u(n) decompose as

$$x(n) = x_f(n)+x_s(t) \tag{37a}$$

$$z(n) = z_f(n)+z_s(t) \tag{37b}$$

$$y(n) = y_f(n)+y_s(t) \tag{37c}$$

and

$$u(n) = u_f(n)+u_s(t). \tag{37d}$$

Substituting (37) in (1), taking the limit $\varepsilon \to 0$ and using (32b) and (32c) we get

$$z_f(n+1) = Sz_f(n)+Gu_f(n) \tag{38a}$$

$$y_f(n) = D_2 z_f(n)+Mu_f(n) \tag{38b}$$

We define a fast performance index J_f as

$$J_f = \sum_{n=0}^{\infty} [y_f^T(n)y_f(n)+u_f^T(n)Ru_f(n)]. \tag{39}$$

The fast discrete-time regulator problem (38), (39) is a standard problem and it is well-known [4] that if the triple (S,G,D_2) is stabilizable-detectable (condition b), then the optimal solution is given by

$$u_f(n) = [R+M^TM+G^TP_fG]^{-1}[G^TP_fS+M^TD_2]z_f(n) \triangleq -F_f z_f(n) \tag{40}$$

where P_f is the unique positive semidefinite stabilizing solution of the discrete-time algebraic Riccati equation

$$P_f = Q_f+S^TP_fS-[G^TP_fS+M^TD_2]^T[R+M^TM+G^TP_fG]^{-1} \quad [G^TP_fS+M^TD_2]. \tag{41}$$

Inspection of (12) and (41), together with the uniqueness of the solution of the Riccati equation, shows that

$$P_3(0) = P_f. \tag{42}$$

Motivated by the results of Chow and Kokotovic [2] in the continuous-time case it is natural to ask the question: is there a similar relation between $P_1(0)$ and P_s? The answer is yes, as it can be seen from the following lemma.

Lemma: The matrices $\hat{A}$, $\hat{B}$ and $\hat{Q}$ appearing in (15) are given by

$$\hat{A} = A_s-B_sR_s^{-1}D_s^TC_s$$

$$\hat{B} = B_s$$

$$\hat{Q} = C_s^T(I-D_sR_s^{-1}D_s^T)C_s.$$

The proof of the lemma involves lengthy algebraic manipulations and is omitted due to space limitation.

As a consequence of the lemma we have

$$P_1(0) = P_s \tag{43}$$

Also, condition (a) is equivalent to condition (a´) which is independent of $P_3(0)$. Therefore Theorems 1 and 2 hold under condition (a´) and (b).

Composite Control: With the solutions of the slow and fast problems in hand, a composite feedback control is formed as

$$\begin{aligned} u_c &= u_s + u_f \\ &= -F_s x_s(t) - F_f z_f(n) \end{aligned}$$

Approximating $z_f(n)$ by $z(n)-\overline{z}(t)$, expressing $\overline{z}(t)$ in terms of $x_s(t)$ and $u_s(t)$ and approximating $x_s(t)$ by $x(n)$, we get

$$\begin{aligned} u_c &= F_s x(n) - F_f[z(n) - (I-S)^{-1}(C-GF_s)x(N)] \\ &= -[F_s + F_f(I-S)^{-1}(C-GF_s)]x(n) - F_f z(n) \end{aligned} \tag{44}$$

The composite feedback control law (44) is near-optimal as established in the following theorem.

Theorem 3: Under Conditions (a´) and (b) the composite control (44) is $O(\varepsilon^2)$ near-optimal in the sense that

$$J - J_{opt} = O(\varepsilon^2)$$

The main step in the proof is verifying that the feedback coefficients in (44) are $O(\varepsilon)$ perturbations of the feedback coefficients in the optimal control (3), which is not difficult to show. The rest of the proof is similar to the proof of Theorem 2.

APPENDIX A

The functions f_1, f_2 and f_3 in (7)-(9) are defined by

$$\begin{aligned} f_1 = {} & P_1A + P_2C + A^TP_1 + C^TP_3 + C^TP_3C + D_1{}^TD_1 \\ & + \varepsilon[A^TP_1A + A^TP_2C + C^TP_2{}^TA] \\ & - [P_1F + P_2G + C^TP_3G + D_1{}^TM + \varepsilon(A^TP_1F + A^TP_2G + C^TP_2{}^TF)] \times \\ & [R + M^TM + G^TP_3G + \varepsilon(F^TP_1F + F^TP_2G + G^TP_2{}^TF)]^{-1} \times \\ & [F^TP_1 + G^TP_2{}^T + G^TP_3C + M^TD_1 + \varepsilon(F^TP_1A + G^TP_2{}^TA + F^TP_2C)], \end{aligned}$$

$$\begin{aligned} f_2 = {} & P_1B + P_2S + C^TP_3S + D_1{}^TD_2 + \varepsilon[A^TP_1B + A^TP_2S + C^TP_2{}^TB] \\ & - [P_1F + P_2G + C^TP_3G + D_1{}^TM + \varepsilon(A^TP_1F + A^TP_2G + C^TP_2{}^TF)] \times \\ & [R + M^TM + G^TP_3G + \varepsilon(F^TP_1F + F^TP_2G + G^TP_2{}^TF)]^{-1} \times \end{aligned}$$

$$[G^TP_3S+M^TD_2+\varepsilon(F^TP_1B+G^TP_2{}^TB+F^TP_2S)]\ ,$$

and

$$\begin{aligned} f_3 = {} & S^TP_3S+D_2{}^TD_2+\varepsilon[B^TP_1B+B^TP_2S+S^TP_2{}^TS] \\ & -[S^TP_3G+D_2{}^TM+\varepsilon(B^TP_1F+B^TP_2G+S^TP_2{}^TF)]\ \text{x} \\ & [R+M^TM+G^TP_3G+\varepsilon(F^TP_1F+F^TP_2G+G^TP_2{}^TF)]^{-1}\ \text{x} \\ & [G^TP_3S+M^TD_2+\varepsilon(F^TP_1B+G^TP_2B+F^TP_2S]\ . \end{aligned}$$

APPENDIX B

Stability Criteria

Consider the following discrete system

$$\begin{cases} x(n+1) = (I+\varepsilon A)x(n)+\varepsilon BZ(n) & \text{(1-a)} \\ Z(n+1) = Cx(n)+SZ(n) & \text{(1-b)} \end{cases}$$

where I-S is invertible.

Using the transformation

$$Y(n) = Z(n)+Lx(n)$$

we have

$$x(n+1) = [I+\varepsilon A-\varepsilon BL]x(n) + \varepsilon BY(n) \tag{2-a}$$

$$Y(n+1) = [C-SL+L+\varepsilon LA-\varepsilon LBL]x(n) + [S+\varepsilon LB]Y(n) \tag{2-b}$$

It can be shown that for sufficiently small ε there exists L such that $C-SL+L+\varepsilon LA-\varepsilon LBL = 0$ and it can be approximated by $L = -(I-S)^{-1}C+O(\varepsilon)$ which reduces the above system to the following triangular form

$$\begin{bmatrix} (n+1) \\ Y(n+1) \end{bmatrix} = \begin{bmatrix} J+\varepsilon\tilde{A}+O(\varepsilon^2) & \varepsilon B \\ 0 & S+O(\varepsilon) \end{bmatrix} \begin{bmatrix} (n) \\ Y(n) \end{bmatrix} \tag{3}$$

Now, if $\tilde{A}$ is stable in continuous-time sense; i.e., its eigenvalues have negative real parts, and S is stable in discrete-time sense; i.e., $|\lambda(s)|<1$, then, for sufficiently small ε, $I+\varepsilon\tilde{A}+O(\varepsilon^2)$ and $S+O(\varepsilon)$ have eigenvalues inside the unit circle which implies the stability of system (3) and therefore (1).

REFERENCES

1. G. Blankenship, "Singularly Perturbed Difference Equations in Optimal Control Problems," IEEE Trans. Automatic Control, Vol. AC-26, No. 4, Aug. 1981.

2. J.H. Chow and P.V. Kokotovic, "A Decomposition of Near-Optimum Regulators for Systems with Slow and Fast Modes," IEEE Trans. Automatic Control, Oct. 1976.

3. P. Dorato and A.H. Levis, "Optimal Linear Regulators: The Discrete-Time Case," IEEE Trans. Automatic Control, Dec. 1971.

4. H. Kwakernaak and R. Sivan, "Linear Optimal Control Systems," Wiley-Interscience, 1972.

APPLICATIONS OF A STABILITY CRITERION FOR SINGULARLY PERTURBED SYSTEMS

ALI SABERI
Department of Electrical Engineering and System Science
Michigan State University
East Lansing, MI 48824

ABSTRACT

In this paper the stability criterion of singularly perturbed systems reported in [6] is employed to study transient stability of a synchronous generator connected to an infinite bus. The effect of field-flux decay is included in the model of the generator.

INTRODUCTION

Stability properties of singularly perturbed systems have been investigated by several authors over the past two decades. (See [1] for a survey.) In [2-5] Lyapunov methods have been employed. In a recent paper [6] a new stability criterion for nonlinear singularly perturbed systems has been suggested. In [6] it is shown that a quadratic-type Lyapunov function for a singularly perturbed system can be obtained as a weighted sum of quadratic-type Lyapunov functions for two lower order systems, the reduced-order system and boundary-layer system.

In this paper we use the stability criterion of [6] to study the stability of a synchronous generator connected to an infinite bus.

The paper is organized as follows. In section 1 we briefly state the main result of [6]. In section 2 we apply the stability criterion of [6] to the transient stability problem of a power system.

1. Stability Result

Consider the nonlinear singularly perturbed system

$$\begin{aligned} \dot{x} &= f(x,y) \qquad x \in B_x \subset R^n \ * \\ \varepsilon\dot{y} &= g(x,y) \qquad y \in B_y \subset R^m \end{aligned} \tag{1}$$

We assume that, in B_x and B_y, the origin ($x = 0$, $y = 0$) is the unique equilibrium point and (1) has a unique solution. A reduced-order system is defined by setting $\varepsilon = 0$ in (1) to obtain

$$\dot{x} = f(x,y) \tag{2a}$$

$$0 = g(x,y) \tag{2b}$$

Assuming that, in B_x and B_y, (2b) has a unique root $y = h(x)$, the reduced-order system is rewritten as,

$$\dot{x} = f(x,h(x)) \triangleq f_r(x) \tag{3}$$

A boundary layer system is defined as:

$$\frac{dy}{d\tau} = g(x,y) \tag{4}$$

---

[*] The symbol B_x indicates a closed sphere centered at $x = 0$.

where $\tau = t/\varepsilon$ is a stretching time scale. In (4) the vector $x\epsilon R^n$ is treated as a fixed unknown parameter that takes values in B_x. Let the following condition hold.

(I) The reduced system (3) has a Lyapunov function $V:R^n \to R_+$ such that for all $x\epsilon B_x$

$$(\nabla_x V(x))^T f_r(x) \leq -\alpha_1 \psi^2(x), \quad \alpha_1 > 0$$

where $\psi(\cdot)$ is a positive definite function.

(II) The boundary layer system (4) has a Lyapunov function $W(x,y):R^n \times R^m \to R_+$ such that for all $x\epsilon B_x$ and $y\epsilon B_y$

$$(\nabla_y W(x,y))^T g(x,y) \leq -\alpha_2 \phi^2(y - h(x)), \quad \alpha_2 > 0$$

where $\phi(\cdot)$ is a positive definite function.

(III) The following three inequalities hold for any $x\epsilon B_x$ and $y\epsilon B_y$

(a) $(\nabla_x W(x,y))^T f(x,y) \leq \gamma\phi^2(y - h(x)) + \beta_2\psi(x)\phi(y - h(x))$

(b) $(\nabla_x V(x))^T (f(x,y) - f(x,h(x))) \leq \beta_1\psi(x)\phi(y - h(x))$

where β_1, β_2 and γ are nonnegative constants.

The stability criterion is stated in the following theorem:

Theorem 1: Suppose that Condition (I)-(III) hold; let d be a positive number such that $0 < d < 1$ and let $\varepsilon^*(d)$ be the positive number given by

$$\varepsilon^*(d) = \frac{\alpha_1\alpha_2}{\alpha_1\gamma + [\beta_1(1 - d) + \beta_2 d]^2/4d(1 - d)}$$

Then for all $\varepsilon < \varepsilon^*(d)$, the origin is an asymptotic equilibrium of (1) and

$$\nu(x,y) = (1 - d)V(x) + dW(x,y)$$

is a Lyapunov function for (1).

2. APPLICATION TO TRANSIENT STABILITY OF A SYNCHRONOUS GENERATOR CONNECTED TO AN INFINITE BUS

There has been quite a bit of literature on the use of Lyapunov's method to analyze transient stability of power systems (for a survey see [7] and for recent developments see [8]). In the standard approach, a generator is represented by a second-order model, usually referred to as the classical model, where the state variables are δ and $\omega = \dot{\delta}$; δ being the angle between the generated voltage and some reference. The need for higher models has been recognized [7] and several studies that employ higher order models have been conducted. Most of these studies are restricted to one machine connected to an infinite bus. The simplest higher order model that can be considered is a third-order model, known as the one-axis model, in which a field-flux decay is taken into consideration. Saddiquee [9] (see also [10]) studied the stability of one generator connected to an infinite bus using the one-axis model and came up with a Lyapunov function that can be used to estimate the domain of attraction. More recently, Sasaki [11] criticized Saddiquee's work on the basis that it did not take into consideration that the state variables have different speeds, and proposed an alternative approach. Between the work of Saddiquee and the criticism of Sasaki we found an interesting problem where the use of singular perturbations may be useful.

We consider a synchronous generator connected to an infinite bus through a transmission line (Fig. (1)).

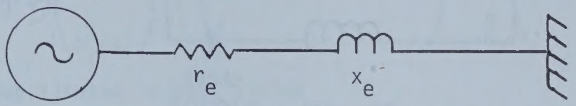


Fig. (1)

The one-axis model (see [9], [7] or [11]) is given by

$$\tau'_{do}\dot{E}'_q = -\frac{(x_d + x_e)}{(x'_d + x_e)} E'_q + \frac{(x_d - x'_d)}{(x'_d + x_e)} V \cos \delta + E_{FD}, \tag{5a}$$

$$\dot{\delta} = \omega, \tag{5b}$$

$$M\dot{\omega} = -D\omega + P_m - \frac{E'_q V}{(x'_d + x_e)} \sin \delta, \tag{5c}$$

where

τ'_{do} = direct-axis open circuit transient time constants;
x_d = direct-axis synchronous reactance of the generator;
x'_d = direct-axis transient reactance of the generator;
E'_q = instantaneous voltage proportional to field-flux linkage;
δ = angle between voltage of infinite bus and E'_q;
D = damping coefficients of the generator;
E_{FD} = field voltage (assumed constant);
P_m = mechanical input to the generator (assumed constant);
ω = slip velocity of the generator;
x_e = reactance of the transmission line;
V = voltage of infinite bus.

In writing equation (5) we have employed some standard assumptions, namely, that the transmission line resistance is negligible, a second order harmonic term in the right hand side of (5c), which results from saliency, is negligible and that $x'_d \simeq x'_q$ (q-axis synchronous reactance). Had the field-flux decay been neglected, E'_q would have been constant and equations (5b) and (5c) would give the classical model. Saddique [9] found a Lyapunov function, which will be given later in this section, and used it to estimate the domain of attraction. The knowlege of the domain of attraction can then be used to estimate the critical clearance time, which we briefly define by means of a simple example. Consider Fig. (2) in which a single machine is connected to an infinite bus through two parallel transmission lines A and B. Suppose that a short circuit fault takes place on line B. Equation (5) is written for the post-fault system with line B disconnected and an estimate of the domain of attraction is obtained. Trajectories of the faulted system are obtained by integrating equations describing the faulted system starting at the pre-fault conditions. The critical clearance time is taken as the time when the trajectories of the faulted system hit the boundary of the estimated domain of attraction. The idea is that if the fault is cleared any time before the critical clearance time, the state of the system will be inside the domain of attraction, hence the system will reach equilibrium.

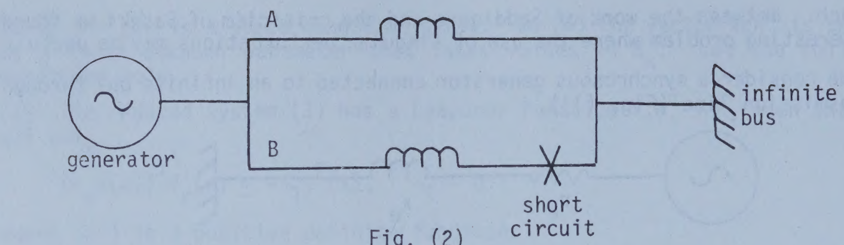


Fig. (2)

Sasaki's criticism of Saddiquee's work was based on the observation that the time constant τ'_{do} is large relative to the critical clearance time (typically, τ'_{do} is greater than five seconds while the critical clearance time is less than one second). Therefore, E'_q will move slowly and changes in E'_q during the fault will be small (typically, less than 0.3 p.u.). He suggested that treating E'_q as a state variable in estimating the domain of attraction might not be appropriate since extending the domain of attraction in the direction of the E'_q-axis, which would be on the expense of its extension in the direction of the δ-axis, would not be beneficial in computing the critical clearance time. He has proposed an alternative method in which equations (5b) and (5c) are used as a model for the machine but with E'_q treated as a varying parameter. He, then, gave a Lyapunov function that should work for all the values of E'_q and used it to estimate the domain of attraction. Sasaki's method, however, is not theoretically justified because he overlooked that changes in E'_q would result in changes in the equilibrium point of (5b) and (5c) and did not take those changes into consideration. It is interesting, however, that Sasaki's viewpoint of treating the slow variable as a varying parameter is identical to the way the boundary-layer system is defined in our singular perturbation method.

We will apply the stability criterion of section (1) to study the stability of the equilibrium of (5). The first task is to bring (5) into the singularly perturbed form. Let $\bar{E}'_q$, $\bar{\delta}$ and $\bar{\omega}$ denote the equilibrium of (5). The state variables are taken as $x = E'_q - \bar{E}'_q$, $y_1 = \delta - \bar{\delta}$ and $y_2 = \omega - \bar{\omega}$, so that x will be the slow variable. In order to have the singularly perturbed form (1), we define ε as $\varepsilon = 1/\tau'_{do}$ and change the time scale from t to t/τ'_{do} to obtain

$$\dot{x} = -ax + b[\cos(y_1 + \bar{\delta}) - \cos\bar{\delta}] \triangleq f(x,y) \tag{6a}$$

$$\varepsilon\dot{y}_1 = y_2 \triangleq g_1(x,y) \tag{6b}$$

$$\varepsilon\dot{y}_2 = -\lambda y_2 - c[(1+x)\sin(y_1 + \bar{\delta}) - \sin\bar{\delta}] \triangleq g_2(x,y) \tag{6c}$$

where

$$a = \frac{x_e + x_d}{x_e + x'_d},\quad b = \frac{x_d - x'_d}{x_e + x'_d},\quad c = \frac{1}{M(x_e + x'_d)},\quad \lambda = \frac{D}{M}.$$

At this point it is not difficult to obtain Lyapunov functions V(x) and W(x,y) that satisfy Condition (I)-(III) of Theorem 1 so that our stability criterion holds for sufficient small ε. The difficult task is that ε is specified in advance and that the upper bound given by Theorem 1 should be greater than ε. That requires great carefulness in choosing V and W and in satisfying the corresponding inequalities in order to get an upper bound which is not very conservative. The reduced system is obtained by setting $\varepsilon = 0$ in (6) to get

$$y = h(x) = \begin{pmatrix} h_1(x) \\ \\ h_2(x) \end{pmatrix} = \begin{pmatrix} \sin^{-1}\left(\frac{\sin \bar{\delta}}{1+x}\right) - \bar{\delta} \\ \\ 0 \end{pmatrix} \tag{7}$$

which when substituted in (6a) yields

$$x = -ax + b[\cos(h_1(x) + \bar{\delta}) - \cos \bar{\delta}] \triangleq -ax + bN(x) \tag{8}$$

The boundary-layer system is defined by

$$\frac{dy_1}{d\tau} = y_2, \tag{9a}$$

$$\frac{dy_2}{d\tau} = -\lambda y_2 + c[(1 + x)\sin(y_1 + \bar{\delta}) - \sin \bar{\delta}] \tag{9b}$$

In order for the reduced and boundary-layer systems to be well defined, as well as for other reasons that will be apparent later, we restrict x and y_1 to a region defined by the following inequalities

$$x \geq -\theta \tag{10a}$$

$$-(\pi + h_1(x) + 2\bar{\delta}) \leq y_1 \leq \pi - (h_1(x) + 2\bar{\delta}) \tag{10b}$$

$$[y_1 - h_1(x)][\sin(y_1 + \bar{\delta}) - \sin(h_1(x) + \bar{\delta})] \geq \eta |y_1 - h_1(x)|^2 \tag{10c}$$

where η and θ are positive numbers to be specified later; should satisfy

$$\left|\frac{\sin \bar{\delta}}{1 - \theta}\right| < 1$$

To study the stability of the reduced system (8) we choose

$$v(x) = \int_0^x [a\sigma - bN(\sigma)]\, d\sigma \tag{11}$$

as a Lyapunov function. From the definition of $N(x)$ it follows that $N(0) = 0$, $xN(x) > 0$ and

$$\left|\frac{\partial N}{\partial x}\right| = \left| \frac{1}{\sqrt{(1 + x)^2 - (\sin \bar{\delta})^2}} \left(\frac{\sin \bar{\delta}}{1 + x}\right)^2 \right| \leq K_N$$

where

$$K_N = \frac{1}{\sqrt{(1 - \theta)^2 - (\sin \bar{\delta})^2}} \left(\frac{\sin \bar{\delta}}{1 - \theta}\right)^2.$$

We assume that the positive number θ can be chosen such that $K_N < (\frac{a}{b})$. It follows that $V(x)$ is positive-definite. The derivative of $V(x)$ along the trajectory of (8) is given by

$$\dot{V} = \frac{\partial V}{\partial x}(-ax + bN(x)) = -(ax - bN(x))^2.$$

Therefore, Condition (I) is satisfied with $\psi(x) = |ax - bN(x)|$ and $\alpha_1 = 1$. Next, we consider the boundary-layer system (9). This system has equilibrium at $y_1 = h_1(x)$ and $y_2 = 0$. We choose a Lure-type Lyapunov function $W(x,y)$ given by

$$W(x,y) = \frac{\lambda}{2}(y_1 - h_1(x))^2 + y_2(y_1 - h_1(x)) + \frac{n}{2\lambda} y_2^2 + \frac{nc}{\lambda}(1 + x)\int_{h_1(x)}^{y_1} M_x(\sigma)d\sigma \qquad (12)$$

where $n > 1$ and

$$M_x(\sigma) = \sin(\sigma + \bar{\delta}) - \sin(h_1(x) + \bar{\delta}).$$

Inequalities (10a) and (10b) guarantee that $W(x,y)$ is positive-definite. The derivative of $W(x,y)$ along the trajectories of (9) is given by

$$\frac{dW}{d\tau} = (\nabla_y W(x,y))^T g(x,y) = -(n - 1)y_2^2 - c(1 + x)(y_1 - h_1(x))M_x(y_1).$$

Using inequality (10c) we get

$$\frac{dW}{d\tau} \leq -(n - 1)y_2^2 - c(1 - \theta)\eta|y_1 - h_1(x)|^2$$

A comparison function for the boundary-layer system is taken as

$$\phi(y - h(x)) = \begin{pmatrix} |y_1 - h_1(x)| \\ \xi|y_2| \end{pmatrix}$$

where ξ is an arbitrary positive number that will play an important role in improving the upper bound. With the above choice of the comparison function and with the choice

$$n = 1 + c(1 - \theta)\eta\xi^2,$$

Condition (II) is satisfied with $\alpha_2 = c(1 - \theta)\eta$.

It remains now to verify inequalities (III-a) and (III-b).

$$\frac{\partial}{\partial x} W(x,y)f(x,y) = [-(ax - bN(x)) + b(\cos(y_1 + \bar{\delta}) - \cos(h_1(x) + \bar{\delta}))].$$

$$[-\lambda\frac{\partial h_1}{\partial x}(y_1 - h_1(x)) - y_2\frac{\partial h_1}{\partial x} - \frac{nc}{\lambda}(\cos(y_1 + \bar{\delta}) - \cos(h_1(x) + \bar{\delta}))]$$

Using the inequality $\cos(y_1 + \bar{\delta}) - \cos(h_1(x) + \bar{\delta}) \leq |y_1 - h_1(x)|$, we get

$$\begin{aligned}\frac{\partial W}{\partial x} f(x,y) &\leq \lambda K_1\psi(x)|y_1 - h_1(x)| + K_1\psi(x)|y_2| + \frac{nc}{\lambda}\psi(x)|y_1 - h_1(x)| \\ &\quad + bK_1|y_1 - h_1(x)||y_2| + bK_1|y_1 - h_1(x)|^2 \\ &\leq \gamma\,\phi^2(y - h(x)) + \beta_2\psi(x)\phi(y - h(x))\end{aligned}$$

where

$$\gamma = \frac{1}{\xi}\left(\frac{1}{2} bK_1 + bK_1\lambda\xi\right),$$

$$\beta_2 = \frac{1}{\xi}\sqrt{K_1^2 + \xi^2\left(\lambda K_1 + \frac{nc}{\lambda}\right)^2},$$

and K_1 is an upper bound on $\frac{\partial h_1}{\partial x}$ (i.e., $|\frac{\partial h_1}{\partial x}| \leq K_1$), which is given by

$$K_1 = \frac{\sin\bar{\delta}}{(1 - \theta)\sqrt{(1 - \theta)^2 - (\sin\bar{\delta})^2}}$$

So, inequality (III-a) holds. Finally,

$$\frac{\partial V}{\partial x}\,(f(x,y) - f(x,h(x))) = b(ax - bN(x))(\cos(y_1 + \overline{\delta}) - \cos(h_1(x) + \overline{\delta}))$$

$$b\psi(x)\,|y_1 - h_1(x))|$$

$$b\psi(x)\,\phi(y - h(x))$$

Thus, all the conditions of Theorem 1 hold and we conclude that the equilibrium point of (5) is asymptotically stable for all $\varepsilon < \varepsilon^{*}(d)$ for any $d\varepsilon(0,1)$ where

$$\varepsilon^{*}(d) = \frac{c(1-\theta)\eta\xi}{\frac{1}{2}\,bK_1 + bK_1\lambda\xi + \frac{1}{4d(1-d)\xi}\left[b(1-d)\xi + d\sqrt{K_1^2 + \xi^2(\lambda K_1 + \frac{\eta c}{\lambda})^2}\right]^2}. \tag{13}$$

Moreover, $\nu(x,y)$, given by

$$\nu(x,y) = (1-d)\int_0^x [a\sigma - bN(\sigma)]\,d\sigma + d[\,\frac{\lambda}{2}\,(y_1 - h_1(x))^2 + y_2(y_1 - h_1(x))$$

$$+ \frac{\eta}{2\lambda}\,y_2^2 + \frac{\eta c}{\lambda}\,(1 + x)\int_{h_1(x)}^{y_1} M_x(\sigma)\,d\sigma], \tag{14}$$

is a Lyapunov function.

In (13) and (14), there are four unspecified positive parameters, namely, d, θ, η and ξ. The first three should be chosen to satisfy certain restrictions that were imposed throughout the derivation, while the fourth parameter ξ is arbitrary. A reasonable procedure for choosing these parameters is to start by choosing d. Since we are interested in extending the domain of attraction in the direction of the y-axis on the expense of its extension in the direction of the x-axis, we will be interested in choosing small values of d. With d chosen, we proceed to choose θ and η. The final step is the choice of the arbitrary parameter ξ. In making these choices we are guided by the requirement that $\varepsilon^{*}(d)$ (given by (13)) should be greater than ε, and by our desire to obtain a large estimate of the domain of attraction. Several trials may be needed before an acceptable choice is reached. Let us illustrate that by a numerical example. Consider the numerical data used by Saddiquee [9]:

$M = 147 \times 10^{-4}$ power second squared per radian,

$x_d' = 0.3$ p.u., $x_e = 0.4$ p.u., $x_d = 1.15$ p.u.,

$\tau_{do}' = 6.6$ sec., $P_m = .815$ p.u., and $E_{FD} = 1.22$ p.u.

From these data we get

$a = 2.7$, $b = 1.7$, $c = \frac{2 \times 10^4}{147}$,

$\overline{E}_q' = 1$ p.u., $\overline{\delta} = 0.42$ rad, $\varepsilon = \frac{1}{6.6} = 0.1515.$

In addition we choose $\lambda = \frac{D}{M} = 4.$

Equation (5) is given by

$$\dot{x} = -2.7x + 1.7[\cos(y_1 + 0.42) - \cos 0.42] \tag{15a}$$

$$\varepsilon \dot{y}_1 = y_2 \tag{15b}$$

$$\varepsilon \dot{y}_2 = -4y_2 - \frac{2 \times 10^4}{147} [(1 + x)\sin(y_1 + 0.42) - \sin 0.42] \tag{15c}$$

We observe that (15), beside having a stable equilibrium point at the origin, has two unstable equilibrium points A and B, given by

$$A = (0,\ 2.3016,\ 0),$$

$$B = (-0.5932,\ 1.18,\ 0).$$

We start by choosing $d = 0.01$, $\theta = 0.42$ and $\eta = 0.35$. With this choice $\varepsilon^*(d)$ is given by

$$\varepsilon^*(0.01) = \frac{27.619\xi}{1.4488+11.59\xi+(25.25/\xi)\left[1.683\xi+.01\sqrt{2.9051+\xi^2(6.8178+34n)^2}\right]^2}$$

The choice $\xi = 0.1$ yields $\varepsilon^*(0.01) = 0.18440$ which is acceptable since $\varepsilon = 0.1515$. Now, $n = 1.2762$ and $\nu(x,y)$ is given by

$$\nu(x,y) = 0.99\ V(x) + 0.01\ W(x,y).$$

An estimate of the domain of attraction can be obtained by means of closed Lyapunov surfaces. However, a better estimate can be obtained by means of open Lyapunov surfaces [12]. Let us define the points C and D by

$$C = (-\theta,\ -\bar{\delta} + \cos^{-1}(\cos\bar{\delta} - \tfrac{a\theta}{b}),\ 0),$$

$$D = (-\theta,\ -\bar{\delta} - \cos^{-1}(\cos\bar{\delta} - \tfrac{a\theta}{b}),\ 0),$$

and let

$$C_{max} = \min\{\nu(x_A,y_A),\ \nu(x_B,y_B),\ \nu(x_C,y_C),\ \nu(x_D,y_D)\},\ \text{if } \theta > |x_A|, \text{ or}$$

$$C_{max} = \min\{\nu(x_B,y_B),\ \nu(x_C,x_C),\ \nu(x_D,y_D)\},\ \text{if } \theta < |x_A|.$$

then the set $\mathscr{L}$, given by

$$= \{x,y \mid \nu(x,y) \le C_{max} \text{ and } x \ge -\theta\}$$

is included in the domain of attraction. This is verified by verifying that $\mathscr{L}$ is included in the region defined by inequalities (10) and by observing that for any y_1 satisfying

$$-\bar{\delta} - \cos^{-1}(\cos\bar{\delta} - \tfrac{a\theta}{b}) \le y_1 \le -\bar{\delta} + \cos^{-1}(\cos\bar{\delta} - \tfrac{a}{b}),$$

i.e., any y_1 on the line connecting C and D, the derivative x is negative. A sketch of the set $\mathscr{L}$ is shown in Fig. (3); shown also is a sketch of the domain of attraction obtained by Saddiquee's Lyapunov function. From Fig. (3), it is apparent that we succeeded in getting a better domain of attraction in the direction of the y-axis on the expense of reducing its size in the direction of the x-axis. The choice of the parameters d, θ, η and ξ, that we used here, were obtained after several trials. Further improvement is possible.

CONCLUSION

The new stability criterion of [6] was applied to analyze the transient stability of an a-c generator including the effects of flux decay. The analysis and result presented here indicate that singular perturbation decompositions could be quite helpful in the transint stability problem of power systems.

ACKNOWLEDGEMENT

The author thanks Prof. H. Khalil for his guidance during the course of this work.

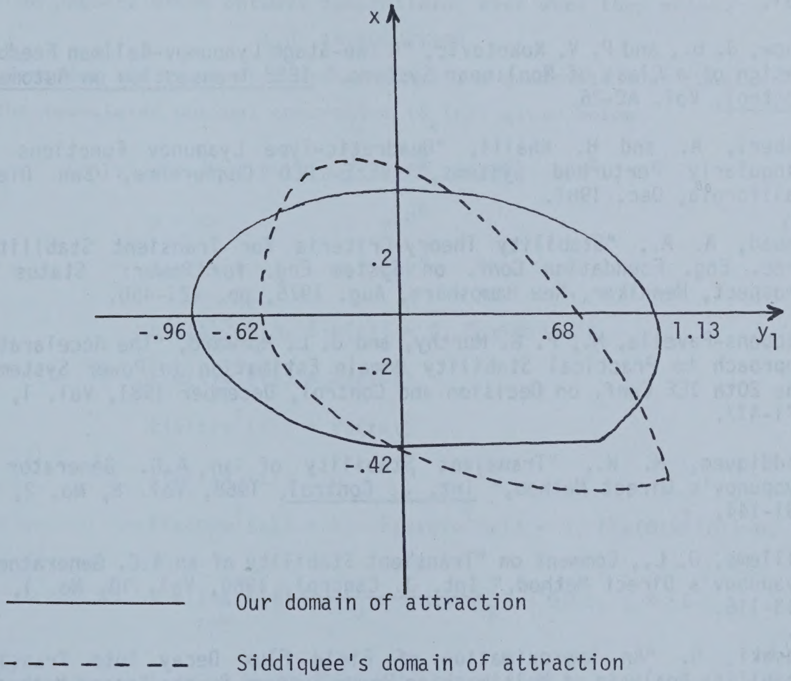


Fig. (3)

REFERENCES

[1] P. V. Kokotovic, R. E. O'Malley Jr. and P. Sannuti, "Singular Perturbation and Order Reduction in Control Theory-An Overview," Automatica, Vol. 12, pp. 123-132.

[2] Habets, P. "Stabilite Asymptotique pour des Problems de Perturbations Singulieres," Centro Internazionalle Matematico Estimo, Ed. Cremonese, 1974.

[3] Grujic, L. T., "Uniform Asymptotic Stability of Nonlinear Singularly Perturbed and Large Scale Systems," Int. J. Control, 1981, Vol. 33, No. 3, pp. 481-504.

[4] Chow, J. H., "Asymptotic Stability of a Class of Nonlinear Singularly Perturbed Systems," J. Franklin Inst., 1978, Vol. 305, No. 5, pp. 275-281.

[5] Chow, J. H., and P. V. Kokotovic, "A Two-Stage Lyapunov-Bellman Feedback Design of a Class of Nonlinear Systems," IEEE Transaction on Automatic Control, Vol. AC-26.

[6] Saberi, A. and H. Khalil, "Quadratic-Type Lyapunov Functions For Singularly Perturbed Systems," IEEE DCD Conference, San Diego, California, Dec. 1981.

[7] Fouad, A. A., "Stability Theory-Criteria for Transient Stability," Proc. Eng. Foundation Conf. on System Eng. for Power: Status and Prospect, Henniker, New Hampshire, Aug. 1975, pp. 421-450.

[8] Ribbens-Pavella, M., P. G. Murthy, and J. L. Horward, "The Accelaration Approach to Practical Stability Domain Estimation in Power Systems," The 20th IEE Conf. on Decision and Control, December 1981, Vol. 1, pp. 471-477.

[9] Siddiquee, M. W., "Transient Stability of an A.C. Generator By Lyapunov's Direct Method," Int. J. Control, 1968, Vol. 8, No. 2, pp. 131-144.

[10] Willems, J. L., Comment on "Transient Stability of an A.C. Generator by Lyapunov's Direct Method," Int. J. Control, 1969, Vol. 10, No. 1, pp. 113-116.

[11] Sasaki, H. "An Approximation of Field Flux Decay Into Transient Stability Analysis of Multimachine Power Systems By the Second Method of Lyapunov," IEEE Transaction on Power Apparatus and Systems, Vol. PAS-98, No. 2, March/April, 1979.

[12] Willems, J. L., "The Computation of Finite Stability Regions By Means of Open Lyapunov Surfaces," Int. J. Control, 1969, Vol. 10, No. 5, pp. 537-544.

BALANCED CONTROLLER REDUCTION

A. YOUSUFF   R.E. SKELTON
Purdue University
School of Aeronautics and Astronautics
West Lafayette, Indiana 47907

Abstract: The standard Linear Quadratic Gaussian (LQG) optimal controller is not minimal in general, even if the plant is minimal. Hence, the order of the optimal controller can be less than that of the plant. This paper shows that the original balancing method of Moore [1] can be used to generate these reduced order optimal controllers, and that the newer balancing methods recently proposed for controller reduction might not generate the reduced order optimal controllers, even when they exist.

1.0 INTRODUCTION

Consider the stochastic system ($\mathcal{S}(n)$), a quadratic cost function ($\mathcal{V}$), and the associated optimal controller ($\mathcal{S}_c(n)$) given below:

$$\mathcal{S}(n):\quad \dot{x} = Ax + Bu + Dw;\quad x\varepsilon\mathcal{R}^n,\quad u\varepsilon\mathcal{R}^m,\quad w\varepsilon\mathcal{R}^d$$

$$y = Cx \qquad ;\quad y\varepsilon\mathcal{R}^k \tag{1.1}$$

$$z = Mx + v \qquad ;\quad z\varepsilon\mathcal{R}^\ell$$

$$\mathcal{E}\{w(t)\} = 0,\ \mathcal{E}\{v(t)\} = 0,\ \mathcal{E}\{x(0)\} = 0$$

$$\mathcal{E}\{w(t)w^T(\tau)\} = W\delta(t-\tau)$$

$$\mathcal{E}\{v(t)v^T(\tau)\} = V\delta(t-\tau)$$

$$\mathcal{E}\{w(t)v^T(\tau)\} = 0$$

$$\mathcal{E}\{x(0)w^T(t)\} = 0,\quad \mathcal{E}\{x(0)v^T(t)\} = 0,\quad \mathcal{E}\{x(0)x^T(0)\}=0,\quad t\geq 0$$

$$\mathcal{V}:\quad \mathcal{V} = \lim_{t\to\infty} \mathcal{E}\{\|y(t)\|_Q^2 + \|u(t)\|_R^2\};\quad Q>0,\quad R>0 \tag{1.2}$$

$$\mathcal{S}_c(n):\quad \dot{\hat{x}} = A_c\hat{x} + Fz\ ;\quad \hat{x}\varepsilon\mathcal{R}^n \tag{1.3a}$$

$$u = G\hat{x}$$

where

$$A_c \triangleq A + BG - FM \tag{1.3b}$$

$$G = -R^{-1}B^TK \tag{1.3c}$$

$$F = PM^TV^{-1}$$

with K and P satisfying the following Riccati equations:

$$KA + A^TK - KBR^{-1}B^TK + C^TQC = 0 \qquad (1.3e)$$

$$PA^T + AP - PM^TV^{-1}MP + DWD^T = 0 \qquad (1.3f)$$

Assume that $S(n)$ is minimal in the sense of Kalman, i.e., the following four conditions hold

a) (A, B) - controllable

b) (A, D) - controllable

c) (C, A) - observable

d) (M, A) - observable.

Then the solutions K and P to (1.3e,f) are positive definite and unique, and hence $S_c(n)$ is the unique controller of order n, [2].

It is tacitly assumed in much of the control literature that this optimal n^{th} order controller $S_c(n)$ is of minimal order, i.e., that a controller of lower order will yield a larger value of the quadratic cost function V. This is not necessarily true. To clarify the above statement, consider the plant $S(n)$ and its controller $S_c(n)$ as two subsystems in series (Fig. 1). Certainly $S(n)$ can be replaced by its minimal realization, and $S_c(n)$ can be replaced by its minimal realization without altering the behavior of the system in Fig. 1. Clearly then, the unobservable, uncontrollable part of $S_c(n)$ can be deleted from $S_c(n)$ without compromising its optimality.

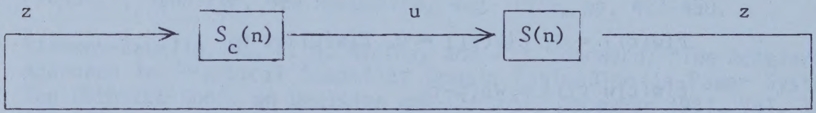


Fig. 1 Series Connection of Subsystems

We will illustrate the 'non-minimality' of $S_c(n)$ with the following example:

$$S(2): \quad A = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad D = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad W = \begin{bmatrix} 1 & -1 \\ -1 & 16 \end{bmatrix}$$

$$C = [2 \quad 0], \quad M = [1 \quad -1], \quad V = Q = R = 1 . \qquad (1.4a)$$

Its optimal controller $S_c(2)$ is described by,

$$\mathcal{S}_c(2):\quad A_C = \begin{bmatrix} -3 & 0 \\ 5 & -4 \end{bmatrix}, \quad F = \begin{bmatrix} 1 \\ -4 \end{bmatrix}, \quad G = [-2 \quad 0] \tag{1.4b}$$

Clearly, $\mathcal{S}_c(2)$ is not observable. Hence, a minimal realization of $\mathcal{S}_c(2)$, denoted by $\mathcal{S}_c^*(1)$, can be obtained by inspection (keeping the observable part of (1.4b)) to produce

$$\mathcal{S}_c^*(1):\quad \begin{aligned} \dot{\hat{x}}_r &= -3\,\hat{x}_r + z \\ u_r &= -2\,\hat{x}_r \ . \end{aligned} \tag{1.5}$$

The evaluation of the cost function V, using either $\mathcal{S}_c(2)$ or $\mathcal{S}_c^*(1)$ as the controller, yields the same value

$$V = 3 \ .$$

Hence $\mathcal{S}_c^*(1)$ is an optimal reduced order controller.

Any controller reduction scheme should at least be able to delete the observable, uncontrollable states from $\mathcal{S}_c(n)$, whenever they exist. The "original" balancing method of Moore [1] (with minor modification) applied to $\mathcal{S}_c(n)$ will always do this, but the "newer" balancing method of Verriest [3], [4], Jonckheere and Silverman [5] will not always do this. The new balancing criteria $P = K = \text{diag}\,\{\pi_1^2, \ldots, \pi_n^2\}$ is proposed in [3-5], where P and K are defined by (1.3e) and (1.3f). In this scheme, states of the optimal n^{th} order controller which are associated with the smaller singular values π_i^2, are deleted to form a reduced order controller. However, since the π_i^2-s are not measures of the observability/controllability of the controller states, this scheme fails to extract a minimal realization of $\mathcal{S}_c(n)$ (denoted by $\mathcal{S}_c^*(r^*)$), even when it exists. To show this, we apply this scheme to the example (1.4) considered previously. This yields the following reduced controller $\mathcal{S}_{c_1}(1)$,

$$\mathcal{S}_{c_1}(1):\quad \begin{aligned} \dot{\hat{x}}_r &= -4\,\hat{x}_r - 3.3636\,z \\ u_r &= 0\,\hat{x}_r \end{aligned} \tag{1.6}$$

which is not a minimal realization of $\mathcal{S}_c(n)$ in (1.4b). Using the controller (1.5) optimality and stability of the closed loop system $\mathcal{S}(2) \oplus \mathcal{S}_c(2)$ is preserved. Using controller (1.6) neither optimality nor stability of the closed loop system is preserved.

The following section shows how the balancing method of [1] may be used to delete the least observable and least controllable states from the full order controller to obtain a reduced controller. In the process, this algorithm extracts the minimal realizations $\mathcal{S}_c^*(r^*)$ when they exist.

2.0 BALANCED CONTROLLER REDUCTION ALGORITHM: (BCRA)

Step 1: Read model data for $\mathcal{S}(n)$, $\mathcal{V}$ as given in (1.1), (1.2).

Step 2: Solve for the standard optimal controller (1.3)

Step 3: If the eigenvalues of A_c satisfy $\lambda_i + \lambda_j \neq 0$, i, j = 1, 2, ..., n, solve the following equations for X and Z.

$$0 = XA_c^T + A_cX + FVF^T \tag{2.1a}$$

$$0 = ZA_c + A_c^TZ + G^TRG \tag{2.1b}$$

where A_c is defined in (1.3b)

Step 4: a) obtain the Spectral Decomposition (SD) of X as

$$X = [U_{11} \quad U_{12}] \begin{bmatrix} \tilde{\Sigma}_1 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} U_{11}^T \\ U_{12}^T \end{bmatrix} \tag{2.2a}$$

$$\text{where } \tilde{\Sigma}_1 = \text{diag}\,\{\sigma_{11}, \sigma_{12}, \ldots, \sigma_{1r_1}\}, \tag{2.2b}$$

$$U_{11} \in R^{nxr_1} \tag{2.2c}$$

b) obtain the SD of $\Sigma_1^{1/2}U_{11}^T Z\, U_{11}\Sigma_1^{1/2}$; $\Sigma_1 \triangleq \text{diag}\{|\sigma_{1i}|\}$, as

$$\Sigma_1^{1/2}\, U_{11}^T Z\, U_{11}\, \Sigma_1^{1/2} = [U_{21} \quad U_{22}] \begin{bmatrix} \tilde{\Sigma}_2 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} U_{21}^T \\ U_{22}^T \end{bmatrix} \tag{2.3a}$$

$$\text{where } \tilde{\Sigma}_2 = \text{diag}\,\{\sigma_{21}, \sigma_{22}, \ldots, \sigma_{2r*}\}, \quad |\sigma_{2,i}| \geq |\sigma_{2,i+1}| \tag{2.3b}$$

$$U_{21} \in R^{r_1xr*} \tag{2.3c}$$

Step 5: Construct the reduced controller $\mathcal{S}_c(r)$ of order $r \leq r^*$ as follows:

$$\begin{aligned} \dot{\hat{x}}_r &= A_r\hat{x}_r + F_r z \\ u_r &= G_r\, \hat{x}_r \end{aligned} \tag{2.4}$$

where

$$A_r \triangleq \Sigma_r^{1/4}\, U_r^T\, \Sigma_1^{-1/2}\, U_{11}^T\, A_c\, U_{11}\, \Sigma_1^{1/2}\, U_r \Sigma_r^{-1/4} \tag{2.5a}$$

$$F_r \triangleq \Sigma_r^{1/4}\, U_r^T\, \Sigma_1^{-1/2}\, U_{11}^T\, F \tag{2.5b}$$

$$G_r \triangleq G U_{11}\, \Sigma_1^{1/2}\, U_r\, \Sigma_r^{-1/4} \tag{2.5c}$$

and where $\Sigma_r = \text{diag}\left\{|\sigma_{21}|, |\sigma_{22}| \ldots, |\sigma_{2r}|\right\} \quad r \leq r^* \leq r_1$

and U_{21} is partitioned as

$$U_{21} = [U_r \ \bar{U}_r], \quad U_r \varepsilon R^{r_1 x r} . \qquad \#$$

Remarks:

i) The value r_1 in (2.2b), which measures the number of non-zero eigenvalues of X is the dimension of the controllable subspace associated with $S_c(n)$, and the columns of U_{11} span this subspace.

ii) The value r^* in (2.3b) is the order of the minimal realization of $S_c(n)$, since r^* measures the dimension of the observable subspace of the completely controllable subspace. Furthermore, the columns of U_{21} form an orthonormal basis for the subspace that is both completely controllable and completely observable.

iii) Clearly, from remark (ii) above, if the desired order r of the controller is r^*, then $S_c(r^*)$ is an optimal reduced controller.

iv) It can be verified that the reduced controller (2.4) is 'balanced', i.e., the following equations hold

$$\tilde{\Sigma}_r^{1/2}\, A_r^T + A_r \tilde{\Sigma}_r^{1/2} + F_r V F_r^T = 0 \tag{2.6a}$$

$$\tilde{\Sigma}_r^{1/2}\, A_r + A_r^T\, \tilde{\Sigma}_r^{1/2} + G_r^T R G_r = 0 \tag{2.6b}$$

Note that the matrix A_c may not be asymptotically stable [6]. Nevertheless, unique solutions to (2.1a) and (2.1b) are guaranteed if and only if A_c has no eigenvalues satisfying $\lambda_i + \lambda_j = 0$. Even if A_c is unstable (provided $\lambda_i + \lambda_j \neq 0$, i, j = 1, 2, ..., n) the values $|\sigma_{2i}|$. still measure the degree of observability/controllability. When A_c

contains eigenvalues such that $\lambda_i + \lambda_j = 0$ for some i and j, a modification in Step 3 of BCRA is required as follows. A result from [7] helps here.

Lemma 1: [7]

a) (A_c, F) *is controllable if and only if* $(A+BG, F)$ *is controllable.*

b) (G, A_c) *is observable if and only if* $(G, A-FM)$ *is observable.*

On the strength of this lemma, when A_c contains eigenvalues satisfying $\lambda_i + \lambda_j = 0$, for some i and j, one may solve the following equations *instead* of (2.1a) and (2.1b),

$$0 = X(A+BG)^T + (A+BG)X + FVF^T \tag{2.7a}$$

$$0 = Z(A-FM) + (A-FM)^T Z + G^T RG , \tag{2.7b}$$

and proceed with the algorithm BCRA without further modification. The algorithm with the (2.7) substitution for (2.1) will be referred to as BCRAM (BCRA modified).

2.1 Examples

Consider the same example in (2.4). Since the controller $S_c(2)$ in (1.4b) is asymptotically stable (hence, eigenvalues of A_c satisfy $\lambda_i + \lambda_j \neq 0$, i, j = 1, 2), equations (2.1a) and (2.1b) can be solved for unique solutions. Using BCRA, the following reduced controller is obtained.

$$S_{c_2}(1): \dot{\hat{x}}_r = -3\,\hat{x}_r + 1.414\ z \qquad u_r = -1.414\ \hat{x}_r \tag{2.8}$$

This controller is a minimal realization of $S_c(2)$ and is related to $S_c^*(1)$ in (1.5) by a similarity transformation. Hence, the algorithm BCRA has served to identify the optimal reduced controller.

One might wonder whether the relative comparisons of the original and new balancing schemes discussed above would be altered if the optimal controller $S_c(n)$ is minimal, i.e. $r^* = n$. In this case both balancing schemes produce reduced controllers which necessarily degrade the performance. In the following example these methods produce stabilizing controllers but the same comparative conclusion as before is illustrated. The three schemes compared are the following:

I. Controller reduction scheme of [3], [4], [5].

II. BCRA (Balanced Controller Reduction Algorithm).

III. BCRAM (BCRA with equations (2.1) replaced by equations (2.7)).

Model, $\mathcal{S}(2)$:

$$\dot{x} = \begin{bmatrix} -1 & 0 \\ 0 & -10 \end{bmatrix} x + \begin{bmatrix} 1 \\ 10 \end{bmatrix} u + \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} w; \quad W = \begin{bmatrix} 3 & 0 \\ 0 & 96 \end{bmatrix}$$

$$y = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} x , \tag{2.9}$$

$$z = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} x + v , \quad V = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

Cost function, $\mathcal{V}$:

$$\mathcal{V} = \lim_{t\to\infty} E\{\|y(t)\|_Q^2 + \|u(t)\|_R^2\} , \ R = 1, \quad Q = \begin{bmatrix} 3 & 10 \\ 10 & 120 \end{bmatrix} \tag{2.10}$$

Full order controller, $\mathcal{S}_c(2)$:

$$\dot{\hat{x}} = \begin{bmatrix} -3 & -10 \\ -10 & -114 \end{bmatrix} \hat{x} + \begin{bmatrix} 1 & 0 \\ 0 & 4 \end{bmatrix} z \tag{2.11}$$

$$u = [-1 \quad -10]\ \hat{x}$$

The reduced controllers obtained are of this form:

$$\mathcal{S}_{c_i}(1): \ \dot{\hat{x}}_{r_i} = A_{r_i} \hat{x}_{r_i} + F_{r_i} z$$

$$i = \text{I, II, III} . \tag{2.12}$$

$$u_{r_i} = G_{r_i} \hat{x}_{r_i}$$

These controllers are compared with respect to the following controller suboptimality index

$$I_i \triangleq \frac{\mathcal{V}_i - \mathcal{V}^*}{\mathcal{V}^*} \tag{2.13}$$

where V^* is the optimal value of cost function (2.10) (i.e. with (2.9) driven by (2.11)), and V_i is the value of the cost function (2.10) when (2.9) is driven by the reduced controller $S_{c_i}(1)$. The results are summarized in Table 1.

Table 1: Comparison of different balancing schemes for controller reduction

Scheme	A_{r_i}	F_{r_i}	G_{r_i}	I_i
I	-114.	[0, 2.8284]	-14.1421	.1201%
II	-113.1813	[.1723, 6.0532]	-6.5434	.0774%
III	-107.7710	[.0108, 10.3900]	-3.6073	.1111%

Evidently, in this example, Scheme II (BCRA) yields the best reduced controller, followed by BCRAM. This example illustrates the fact that when $r < r^*$, ($r=1$, $r^*=2$, in this case), the algorithms BCRA and BCRAM yield different results, and that BCRA is preferred whenever A_c has no eigenvalues satisfying $\lambda_i + \lambda_j = 0$.

3.0 CONCLUSIONS

Since reduced order optimal controllers may exist, the first duty of any controller reduction scheme should be to produce such controllers, whenever they exist. This note has described the balancing scheme of Moore [1] with minor modification, to obtain reduced controllers of any order. These modifications are required to operate on an optimal controller which might be unstable, uncontrollable and/or unobservable. The main attraction of the scheme is its ability to extract the optimal reduced controllers, whenever they exist. It is shown that the newer scheme of balancing [3], [4], [5] applied to the standard LQG problem may not produce optimal reduced controllers even when they exist.

REFERENCES

[1] B.C. Moore; 'Principal Component Analysis in Linear Systems; controllability, observability, and model reduction', *IEEE Trans. Automatic Contr.* Vol. AC-26, pp. 17-32, 1981.

[2] H. Kwakernaak, R. Sivan; *Linear Optimal Control Systems*, Wiley, New Yrok, 1972.

[3] E.I. Verriest; 'Low Sensitivity design and Optimal Order Reduction for the LQG-problem', in *Proc. 24th Symp. Circuits and Systems,* pp. 365-369, June 1981.

[4] E.I. Verriest; 'Suboptimal LQG-design and Balanced Realizations', in *Proc. 20th IEEE, CDC,* pp. 686-687. Dec. 1981.

[5] E.A. Jonckheere and L.M. Silverman; 'A new set of invariants for linear systems - application to approximation', *International Symp. Math. Th. Networks and Syst.,* Santa Monica, CA, 1981.

[6] C.D. Johnson; 'The phenomenon of Homeopathic instability in Dynamical Systems', *Int. J. Contr.,* Vol. 33, No. 1, pp. 159-174, 1981.

[7] W.M. Wonham, *Linear Multivariable Control; A Geometric Approach,* Springer-Verlag, N.Y., 1979.

PSEUDO-BALANCED REALIZATIONS FOR DETERMINISTIC AND STOCHASTIC DISCRETE-TIME SYSTEMS*

R.J. VACCARO and B.W. DICKINSON
Department of Electrical Engineering and Computer Science
Princeton University
Princeton, New Jersey 08544

ABSTRACT

Balanced realizations have proven to be useful tools because of their model reduction properties for both deterministic and stochastic systems. In both cases, balanced realizations are essentially unique (i.e. up to a finite equivalence class). In this paper, a much larger equivalence class of realizations is defined whose elements have the model reduction properties only for the first l reductions of the original model. Such realizations are called *l-pseudo-balanced*.

1. INTRODUCTION

Balanced realizations were developed by Moore [1,2] as an approach to the model reduction problem for deterministic linear systems. Recently, the concept of balanced realizations has been extended to stochastic systems by Desai and Pal [4,5]. The conditions for deterministic balancing are expressed in terms of the controllability and observability Grammians of the system. The analogous conditions for stochastic balancing are expressed in terms of the state covariances of a particular pair of forward and backward innovations models. The analogy between the deterministic and stochastic cases will be explicitly derived. This paper will be restricted to scalar, time-invariant, discrete-time systems.

In the deterministic case, an n^{th} order system $S_n=(A,b,c)_n$ with transfer function $F(z)=c(zI-A)^{-1}b=f(z)/g(z)$ is described by the equations

$$\begin{aligned} x_{k+1}&=Ax_k+bu_k \\ y_k&=cx_k \end{aligned} \tag{1.1}$$

where A is $n \times n$, b is $n \times 1$, and c is $1 \times n$. S_n is said to be a *realization* of $F(z)$; only minimal realizations are considered in this paper, i.e. $\deg(g(z))=n$, f and g coprime. The stochastic case will be developed in Section 4. For any nonsingular matrix T, let the realization $\overline{S}_n=(A_1,b_1,c_1)_n$ be given by

$$A_1=TAT^{-1} \quad , \quad b_1=Tb \quad , \quad c_1=cT^{-1} \tag{1.2}$$

Then S_n and $\overline{S}_n$ are said to be *similar*, and they have the same transfer function. (T is called the *similarity transformation* between S_n and $\overline{S}_n$.)

Balanced realizations are useful for model reduction because of the following two properties:

I. Given an n^{th} order balanced realization $S_n^b=(A_b,b_b,c_b)_n$, where

$$A_b=\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \quad , \quad b_b=\begin{bmatrix} b_1 \\ b_2 \end{bmatrix} \quad , \quad c_b=\begin{bmatrix} c_1 & c_2 \end{bmatrix} \tag{1.3}$$

a "good" reduced-order model of dimension r, $(A,b,c)_r$, is immediately obtained by setting

$$A=A_{11} \quad , \quad b=b_1 \quad , \quad c=c_1 \tag{1.4}$$

---

*This work was supported in part by the National Science Foundation through the Graduate Fellowship Program and grant ENG77-28523.

We will call these reduced order models *partitioned subsystems* of S_n^b.

II. If S_n^b is stable and minimal, then all the partitioned subsytems will be also (i.e. for dim $A_{11}=n-1,n-2,...,1$). This result is true under the assumption of distinct singular values, which will be made later in the paper.

In both the deterministic and stochastic cases, balanced realizations are essentially unique. In this paper it is shown that there are other realizations which satisfy properties I and II, but yet are not balanced according to the conventional definitions. These realizations are now defined.

Definition

Given a transfer function $F(z)$, let S_n^b be a minimal realization in balanced form, and denote its partitioned subsystem of order r by S_r^b. Now let S_n be any other minimal realization of $F(z)$, and denote the rth order partitioned subsystem of S_n by S_r. Then S_n is called an *l-pseudo-balanced realization* if S_r is similar to S_r^b for $r=n-1,n-2,...,n-l$.

Note that for model reduction purposes, l-pseudo-balanced realizations are equivalent to balanced realizations, since their corresponding partitioned subsystems have the same transfer functions (i.e. both classes of realizations yield the same reduced-order models) for the first l partitioned subsystems. Since l-pseudo-balanced realizations are defined by their relationship to balanced ones, we first discuss the balanced case.

2. DETERMINISTIC BALANCED REALIZATIONS

We start with some definitions regarding a realization of $F(z)$, $S_n=(A,b,c)_n$, which we assume to be minimal. The s-observability, t-controllability, and Hankel matrices are given, respectively as

$$O_s=\begin{bmatrix} c \\ cA \\ \cdot \\ \cdot \\ cA^{s-1} \end{bmatrix} , \quad C_t=\begin{bmatrix} b & Ab & \cdots & A^{t-1}b \end{bmatrix} , \quad H_{s,t}=O_s C_t \tag{2.1}$$

The Hankel matrix has rank n for all $s,t \geq n$. If A has all its eigenvalues in $|z|<1$ (i.e. the system is asymptotically stable), then the infinite Hankel matrix is defined as $H=OC$ (i.e. $H_{\infty,\infty}=O_\infty C_\infty$). Finally, the controllability and observability Grammians (over the interval $[0,\infty]$) are given, respectively, as

$$G_c=CC^T \quad , \quad G_o=O^T O \tag{2.2}$$

Since S_n is minimal, G_c and G_o are both positive definite, and so the matrix $G_o G_c$ has strictly positive eigenvalues which we denote σ_i^2 with the ordering $\sigma_1>\sigma_2 \cdots \sigma_n>0$. (In this paper, we will make the simplifying assumption that these eigenvalues are distinct.) Now define the matrix

$$\Sigma=diag(\sigma_1, \cdots, \sigma_n) \tag{2.3}$$

A *balanced* realization is one in which $G_c=G_o=\Sigma$. (Note that Σ is invariant under similarity transformations of S_n. See [1-3] for more details on why this definition leads to good reduced-order models.)

Balancing from a Factorization of the Hankel Matrix

From realization theory, we know that any factorization of the Hankel matrix as $H=OC$ corresponds to a specific minimal realization $(A,b,c)_n$, which is identified as follows: From (2.1) it is clear that

$$c = \textit{first row of } O \tag{2.4}$$

$$b = \textit{first column of } C$$

Now let "↑" and "←" denote the shift-up and shift-left operations, respectively. Then we have

$$OA=\begin{bmatrix} cA \\ cA^2 \\ \cdot \\ \cdot \end{bmatrix}=O^{\uparrow} \tag{2.5}$$

$$AC=\begin{bmatrix} Ab & A^2b & \cdots \end{bmatrix}=\overleftarrow{C}$$

By solving for A in (2.5) we get

$$A=O^{+}O^{\uparrow}=\overleftarrow{C}C^{+} \tag{2.6}$$

where O^{+} is a left inverse of O, and C^{+} is a right inverse of C.

Now look at the singular value decomposition (SVD) of H [6,8]:

$$H=U\Sigma V^T \quad , \quad \Sigma=diag(\sigma_1> \cdots \sigma_n>0) \quad , \quad U^TU=V^TV=I \tag{2.7}$$

(Note that we assume distinct singular values.) Then any grouping of $U\Sigma V^T$ as OC corresponds to some (A,b,c). It is easy to see that the choice

$$C=\Sigma^{1/2}V^T, \quad O=U\Sigma^{1/2} \tag{2.8}$$

means that $G_o=O^TO=G_c=CC^T=\Sigma$, and so this particular factorization of H corresponds to a balanced realization $(A_b,b_b,c_b)_n$. (In this case, the eigenvalues of G_oG_c are obviously $\{\sigma_i^2\}$, and so the Σ obtained by the SVD is the same as the one given in (2.3).) We also see that the only possible nonuniqueness arises from the fact that, for any sign matrix S (a diagonal matrix with ±1 on the diagonal), $O_1=OS$, $C_1=SC$ is another factorization of H which gives a balanced realization $(\bar{A}_b,\bar{b}_b,\bar{c}_b)_n$. In other words, the most general factorization of H that corresponds to a balanced realization is

$$H=(U\Sigma^{1/2}S)(S\Sigma^{1/2}V^T)=O_1C_1 \tag{2.9}$$

for an arbitrary sign matrix S. Using (2.4) and (2.6) we see that $\bar{c}_b=c_bS$, $\bar{b}_b=Sb_b$, $\bar{A}_b=SA_bS$ That is, balanced realizations are unique up to a sign matrix (for distinct σ_i). Another way to view this "uniqueness", however, is that balanced realizations form an equivalence class where the equivalence relation is similarity transformation by a sign matrix. Let this equivalence class be denoted $\{S_n^b\}$. We shall now show that l-pseudo-balanced realizations form a much larger equivalence class which includes $\{S_n^b\}$.

3. DETERMINISTIC PSEUDO-BALANCED REALIZATIONS

Consider an nth order transfer function $F(z)$ and its related Hankel matrix H.

Lemma 1

Given the following factorization, with E an arbitrary nonsingular matrix

$$H=(U\Sigma^{½}E^{-1})(E\Sigma^{½}V^T)=OC \tag{3.1}$$

let the corresponding realizations be $\{S_n(E)\}$ (i.e. the realizations obtained using (2.4) and (2.6)). Then the class of all minimal realizations of $F(z)$ is $\{S_n(E)\}$.

Proof

Let $E=I$. From (2.4) and (2.6)

$$S_n(I)=(A,b,c)=(\Sigma^{½}\overleftarrow{V}^T V\Sigma^{-½},\Sigma^{½}V^T e_1,e_1^T U\Sigma^{½})$$

where $e_1^T=(1,0,\ldots,0)$. Now let $E=E_1$ for some nonsingular matrix E_1. Then

$$S_n(E_1)=(E_1\Sigma^{½}\overleftarrow{V}^T V\Sigma^{-½}E_1^{-1},E_1\Sigma^{½}V^T e_1,e_1^T U\Sigma^{½}E_1^{-1})=(E_1AE_1^{-1},E_1b,cE_1^{-1})$$

From (1.2), $S_n(E_1)$ is similar to $S_n(I)$, and since E_1 is arbitrary, all realizations which are similar to $S_n(I)$ are obtained by proper choice of E_1, and this generates all minimal realizations of $F(z)$.

Note that $\{S_n(E)\}$ is an equivalence class, where the equivalence relation is similarity transformation by an arbitrary nonsingular matrix. Comparing (2.9) with (3.1), it is clear that $\{S_n^b\}$ can be obtained by restricting E to be of the form

$$E=S \tag{3.2}$$

for some sign matrix S; thus $\{S_n(E)\}\supset\{S_n^b\}$. We are now in a position to define $\{l-S_n^{pb}\}$, the class of l-pseudo-balanced realizations. We first need

Definition

An $n \times n$ nonsingular block-diagonal matrix of the form given below, with G a diagonal matrix will be called *l-block-lower-diagonal.*

$$\begin{bmatrix} F & 0 \\ 0 & G_{l\times l} \end{bmatrix} \tag{3.3}$$

Proposition 1

$\{l-S_n^{pb}\}$, the equivalence class of l-pseudo-balanced realizations is that subset of $\{S_n(E)\}$ obtained by restricting E to be an arbitrary l-block-lower-diagonal matrix.

Proof

See Appendix.

From (3.1) and (2.2), we see that for some realization $S_n(E)$,

$$G_c=E\Sigma E^T \quad , \quad G_o=E^{-T}\Sigma E^{-1} \tag{3.4}$$

Using this and Prop. 1, we get

Corollary 1

The necessary and sufficient conditions on the controllability and observability Grammians for the model reduction properties I and II to hold for the first l partitioned subsystems is that (3.4) be satisfied for some l-block-lower-diagonal matrix E.

Comments

(1) The original work on balanced realizations obtained the model reduction properties I and II of Section 1 as a result of the fact that $G_o = G_c = \Sigma$, and so this condition is sufficient for the properties to hold. But from Corollary 1, we see that the necessary and sufficient condition for these properties to hold (for all partitioned subsystems) is just that E be a diagonal matrix. From (3.4), this means that G_c can be an *arbitrary* nonsingular diagonal matrix, and G_o must then be the diagonal matrix given by (3.4).

(2) The computation of balanced realizations does not involve factoring the Hankel matrix (which we have done for ease of exposition). Instead, the similarity transformation T is computed which takes a given system S_n to balanced form. (Note that an eigenvalue problem must be solved to obtain Σ.) An efficient algorithm for computing T is given by Laub [7]. It was hoped that since pseudo-balanced realizations form a larger class than balanced, a more efficient algorithm for computing the similarity transformation T_1 which takes S_n to l-pseudo-balanced form could be found. This does not seem to be possible. The reason is that the condition on G_c and G_o for l-pseudo-balanced realizations (3.4) requires knowledge of Σ (i.e. the eigenvalues of $G_o G_c$), but computation of these eigenvalues is the only "hard" part of Laub's algorithm.

4. THE STOCHASTIC CASE

Consider a stationary, zero-mean, Gaussian stochastic process $\{y_k\}$ which has the covariance function $r_j = E[y_k y_{k+j}^T]$. The necessary and sufficient conditions for a stochastic realization to be balanced (or pseudo-balanced) are expressed in terms of the state covariance matrices of the forward and reversed-time innovations models associated with the covariance function. In order to define these models, the following notation is needed: let

$$Y_k^+ = \begin{bmatrix} y_k \\ y_{k+1} \\ \cdot \\ \cdot \\ \cdot \end{bmatrix}, \qquad Y_k^- = \begin{bmatrix} y_{k-1} \\ y_{k-2} \\ \cdot \\ \cdot \\ \cdot \end{bmatrix} \tag{4.1}$$

and let $\{Y_k^+\}$and$\{Y_k^-\}$ be the Hilbert spaces obtained by taking the closed linear span of the elements in Y_k^+ and Y_k^- respectively. Then the Hankel matrix H and the Toeplitz covariance matrix R are defined as

$$H = E[Y_k^+ (Y_k^-)^T] = \begin{bmatrix} r_1 & r_2 & r_3 & \cdot \\ r_2 & r_3 & r_4 & \cdot \\ r_3 & r_4 & r_5 & \cdot \\ \cdot & \cdot & \cdot & \cdot \end{bmatrix} \tag{4.2}$$

$$R = E[Y_k^+ (Y_k^+)^T] = E[Y_k^- (Y_k^-)^T] = \begin{bmatrix} r_0 & r_1 & r_2 & \cdot \\ r_1 & r_0 & r_1 & \cdot \\ r_2 & r_1 & r_0 & \cdot \\ \cdot & \cdot & \cdot & \cdot \end{bmatrix} = LL^T \tag{4.3}$$

(i.e. - let L be the Cholesky factor of R.) Assume the covariance function is rational so that H has finite rank = n [13]. It will be convenient to work with normalized data defined as follows:

$$\alpha_k^+ = L^{-1} Y_k^+ \qquad \alpha_k^- = L^{-1} Y_k^- \tag{4.4}$$

and it is easy to see that $E[\alpha_k^+(\alpha_k^+)^T]=E[\alpha_k^-(\alpha_k^-)^T]=I$. Define the normalized matrix

$$\overline{H}=E[\alpha_k^+(\alpha_k^-)^T]=L^{-1}HL^{-T}=U\Sigma V^T \tag{4.5}$$

(Note that we will use the singular value decomposition of $\overline{H}$.) We also define the (forward) innovations sequence $\nu_k=y_k-y_k\,|\{Y_k^-\}$, and the reversed-time (backwards) innovations sequence $\gamma_k=y_k-y_k\,|\{Y_{k+1}^-\}$, where "|" denotes orthogonal projection.

By factoring H (as in Section 3), we can obtain a deterministic realization of the output covariance function. The resulting deterministic realization, (F_c,g_c,h_c), will be called the *covariance realization*. From (2.4) and (2.6), we see that for some factorization of H as

$$H=OC \tag{4.6a}$$

with rank O = rank $C = n$,

$$F_c C=\overleftarrow{C} \;\; or \;\; F_c=\overleftarrow{C}C^+ \;\; , \;\; h_c=first\ row\ of\ O \tag{4.6b}$$

The (forward) innovations and reversed-time innovations models are defined as follows:

Innovations Model

$$\begin{aligned} x_{k+1}&=F_i x_k+g_i\nu_k & \qquad E[x_k x_k^T]&=\Pi_i \\ y_k&=h_i x_k+\nu_k & \qquad E[\nu_k\nu_k^T]&=\rho_i \end{aligned} \tag{4.7}$$

Reversed-Time Innovations Model

$$\begin{aligned} z_{k-1}&=F_r z_k+g_r\gamma_k & \qquad E[z_k z_k^T]&=\Pi_r \\ y_k&=h_r z_k+\gamma_k & \qquad E[\gamma_k\gamma_k^T]&=\rho_r \end{aligned} \tag{4.8}$$

(Note that the term "innovations model" refers to a forward innovations model.) For more information on reversed-time models, see [5,11,12,14]. We call (4.7) and (4.8) innovations models *associated* with a covariance function $\{r_j\}$ if the output covariance functions of (4.7) and (4.8) are $\{r_j\}$.

It is well known that the state of an innovations model is a basis for the space $\{Y_k^+\}|\{Y_k^-\}$ [9,10,13], and that this projection is a linear function of Y_k^-, i.e. $x_k=GY_k^-$ for some matrix G. A specific innovations model corresponds to a particular choice of G. For reversed-time models, the state z_k is a basis for the space $\{Y_{k+1}^-\}|\{Y_{k+1}^+\}$, and this projection is a linear function of Y_{k+1}^+. The forward and reversed-time innovations models can both be related to the covariance realization. In order to get the analogy with the deterministic case, the relationships must be (F_i,h_i) = (F_c,h_c) and (F_r,h_r) = (F_c^T,g_c^T). The choices of state variables which give these relationships are shown in the next two Lemmas.

Lemma 2

Let $H=OC$ be the factorization of H given in (4.6). Then an innovations model associated with $\{r_j\}$ will have (F_i,h_i) = (F_c,h_c) if and only if the state is chosen as $x_k=CR^{-1}Y_k^-$. In this case, $\Pi_i=CR^{-T}C^T$.

Proof

See Appendix.

Lemma 3

Let $H=OC$ be the factorization of H given in (4.6). Then the reversed-time innovations model associated with $\{r_j\}$ will have $(F_r,h_r)=(F_c^T,g_c^T)$ if and only if the state is chosen as $z_k=O^TR^{-1}Y_{k+1}^+$. In this case, $\Pi_r=O^TR^{-T}O$.

Proof

Completely analogous to that of Lemma 2.

Consider the factorization of H given in (4.5). After inserting an arbitrary nonsingular matrix E, that factorization can be written as

$$H=(LU\Sigma^{1/2}E^{-1})(E\Sigma^{1/2}V^TL^T)=OC \tag{4.9}$$

Note that the parentheses identify the choice of O and C. This factorization induces a covariance realization by (4.6). Using Lemmas 2 and 3, the state variables and the state covariance matrices of the corresponding forward and reversed-time innovations models are

$$x_x=E\Sigma^{1/2}V^TL^{-1}Y_k^- \qquad z_k=E^{-T}\Sigma^{1/2}U^TL^{-1}Y_k^+ \tag{4.10}$$

$$\Pi_i=E\Sigma E^T \qquad \Pi_r=E^{-T}\Sigma E^{-1} \tag{4.11}$$

Note that choosing a particular E is equivalent to choosing particular state coordinates, and so E parametrizes the class of innovations models corresponding to the given process covariance function (compare Lemma 1).

Comparing (4.11) with (3.4), it is clear that an analogy with the deterministic case has been achieved in which the state covariance matrices of a particular pair of forward and reversed-time innovations models correspond to controllability and observability matrices, respectively. Thus it is not surprising that the definition in [5] of balanced stochastic realizations (developed via other arguments) is the same as the definition of deterministic balanced realizations with G_c, G_o replaced by Π_i, Π_r, and that the algorithm given for computing balancing transformations in the stochastic case is just the Laub algorithm with the same replacement. It is also clear that Proposition 1 and Corollary 1 both apply to the stochastic case as well as the deterministic, and so the equivalence class of l-pseudo-balanced *stochastic* realizations is obtained by restricting E in (4.9) to be an arbitrary l-block-lower-diagonal matrix.

5. APPENDIX

Proof of Proposition 1

Recall that the definition of l-pseudo-balanced realizations requires that their first l partitioned subsystems be similar to those of a balanced realization. The following Lemma relates the partitioned subsystems of any model to a partitioning of O and C.

Lemma A1

Let $S_n=(A,b,c)_n$ be a realization corresponding to a factorization of H as $H=OC$, and let these matrices be partitioned as follows:

$$A=\begin{bmatrix}A_{11} & A_{12}\\ A_{21} & A_{22}\end{bmatrix},\ b=\begin{bmatrix}b_1\\ b_2\end{bmatrix},\ c=\begin{bmatrix}c_1 & c_2\end{bmatrix},\ O=[O_1\ O_2],\ C=\begin{bmatrix}C_1\\ C_2\end{bmatrix} \tag{A.1}$$

where A_{11} is $r\times r$, A_{22} is $(n-r)\times(n-r)$, b_1 is $r\times 1$, c_1 is $1\times r$, O_1 is $\infty\times r$, C_1 is $r\times\infty$. Let O^+ be a left inverse of O which is partitioned as follows:

$$O^+ = \begin{bmatrix} O_1^+ \\ O_2^+ \end{bmatrix} \tag{A.2}$$

Then the partitioned subsystem of order r is given by

$$A_{11} = O_1^+ O_1^{\uparrow}\ ,\quad b_1 = \mathit{first\ column\ of}\ C_1\ ,\quad c_1 = \mathit{first\ row\ of}\ O_1 \tag{A.3}$$

Proof

From (2.5), $A = O^+ O^{\uparrow}$, or

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} = \begin{bmatrix} O_1^+ \\ O_2^+ \end{bmatrix} \begin{bmatrix} O_1^{\uparrow} & O_2^{\uparrow} \end{bmatrix} \tag{A.4}$$

Take the first r rows and columns of (A.3) to obtain $A_{11} = O_1^+ O_1^{\uparrow}$. That b_1 and c_1 are as indicated follows by inspection.

A balanced realization corresponds to the following factorization of H

$$H = \begin{bmatrix} U_1 \Sigma_1^{1/2} & U_2 \Sigma_2^{1/2} \end{bmatrix} \begin{bmatrix} \Sigma_1^{1/2} V_1^T \\ \Sigma_2^{1/2} V_2^T \end{bmatrix} = \begin{bmatrix} O_1 & O_2 \end{bmatrix} \begin{bmatrix} C_1 \\ C_2 \end{bmatrix} \tag{A.5}$$

where

$$O_1 = U_1 \Sigma_1^{1/2}\ ,\quad C_1 = \Sigma_1^{1/2} V_1^T \tag{A.6}$$

(Note - In order for this factorization to correspond to an arbitrary balanced realization, let the sign matrix in (2.9) be absorbed into Σ_1 and Σ_2 so that $\sigma_i^{1/2}$ may be positive or negative numbers.) From Lemma A1, the r^{th} order partitioned subsystem $S_r^b = (A_{11}, b_1, c_1)$ is given by (A.3) with O_1 and C_1 given by (A.6).

Now consider an arbitrary realization $\overline{S}_n$ which is similar to S_n^b. By Lemma 1, $\overline{S}_n$ corresponds to the following factorization of H for some nonsingular matrix E. (Note that E is partitioned as E_{ij} while E^{-1} is partitioned as $\overline{E}_{ij}$.)

$$H = \left\{ \begin{bmatrix} U_1 \Sigma_1^{1/2} & U_2 \Sigma_2^{1/2} \end{bmatrix} \begin{bmatrix} E_1 & E_2 \\ E_3 & E_4 \end{bmatrix} \right\} \left\{ \begin{bmatrix} \overline{E}_1 & \overline{E}_2 \\ \overline{E}_3 & \overline{E}_4 \end{bmatrix} \begin{bmatrix} \Sigma_1^{1/2} V_1^T \\ \Sigma_2^{1/2} V_2^T \end{bmatrix} \right\} = \overline{O}\,\overline{C} = \begin{bmatrix} \overline{O}_1 & \overline{O}_2 \end{bmatrix} \begin{bmatrix} \overline{C}_1 \\ \overline{C}_2 \end{bmatrix} \tag{A.7}$$

where

$$\overline{O}_1 = U_1 \Sigma_1^{1/2} E_1 + U_2 \Sigma_2^{1/2} E_3\ ,\quad \overline{C}_1 = \overline{E}_1 \Sigma_1^{1/2} V_1^T + \overline{E}_2 \Sigma_2^{1/2} V_2^T \tag{A.8}$$

The r^{th} order partitioned subsystem of $\overline{S}_n$ is $\overline{S}_r = \overline{(A, b, c)}_n$, and it is given by (A.3) with $O_1 = \overline{O}_1$, $C_1 = \overline{C}_1$. We can now prove

Lemma A2

> If S_r^b is similar to $\overline{S}_r$ for $r = n-1, n-2, \ldots, n-l$, then the matrix E in (A.7) must be l-block-lower-diagonal.

Proof

Since the realization formula (A.3) uses O_1 and C_1 as if they were observability and controllability matrices respectively (which they are not since their product is not a Hankel matrix in general), it is clear that a well known result from system theory is applicable here - namely, that S_r^b is similar to $\overline{S}_r$ if and only if $\overline{O}_1 = O_1 T$ and $\overline{C}_1 = T^{-1} C_1$ for some nonsingular T [15]. Substituting (A.6) and (A.8) into these equations yields

$$U_1 \Sigma_1^{1/2} E_1 + U_2 \Sigma_2^{1/2} E_3 = U_1 \Sigma_1^{1/2} T \tag{A.9}$$

$$\overline{E}_1 \Sigma_1^{1/2} V_1^T + \overline{E}_2 \Sigma_2^{1/2} V_2^T = T^{-1} \Sigma_1^{1/2} V_1^T \tag{A.10}$$

Multiply (A.9) on the left by U_2^T and (A.10) on the right by V_2, and use the fact that the columns of U and V are orthogonal to obtain

$$E_3 = 0\ ,\quad \overline{E}_2 = 0\ ,\quad \overline{E}_1 = E_1^{-1} \tag{A.11}$$

Since E_3 is zero, E is a block upper-triangular matrix, hence its inverse must also be, and so $\bar{E}_3=0$. Similarly, $\bar{E}_2=0$ implies $E_2=0$. Thus for a fixed r, E must be a block-diagonal matrix. Applying this argument for $r=n-1, n-2, \ldots, n-l$ shows that E is l-block-lower-diagonal.

It is easy to see that the converse of Lemma A2 is also true, since if E is l-block-lower-diagonal, then from (A.9) or (A.10), $T = E_1$ is the similarity transformation between S_r^b and $\bar{S}_r$. Thus l-pseudo-balanced realizations are precisely that subset of $\{S_n(E)\}$ obtained by restricting E to be an l-block-lower-diagonal matrix. It is clear that $\{l-S_n^{pb}\}$ is an equivalence class where the equivalence relation is similarity transformation by an l-block-lower-diagonal matrix.

Proof of Lemma 2

The "if" part of the proof is as follows: note that $x_k|\{Y_k^-\} = x_k$ so that this choice of state does indeed give an innovations model. We need to show that it gives the innovations model with $(F_i, h_i)=(F_c, h_c)$.

$$E[x_{k+1}x_k^T] = E[(F_i x_k + g_i \nu_k)x_k^T]$$

$$CR^{-1}\overleftarrow{R}R^{-T}C^T = F_i E[x_k x_k^T] = F_i \Pi_i$$

$$\overleftarrow{C}R^{-T}C^T = F_i CR^{-1}RR^{-T}C^T$$

$$F_c CR^{-T}C^T = F_i CR^{-T}C^T = F_i \Pi_i \quad (i.e.\ \Pi_i = CR^{-T}C^T) \tag{A.12}$$

Note that since $R=LL^T$, $CR^{-T}C^T=(CL^{-T})(CL^{-T})^T$, and since CL^{-T} has rank n, $CR^{-T}C^T$ is nonsingular. So by (A.12), $F_i=F_c$. Now let $e_1^T=(1,0,0,\ldots)$. Then

$$E[y_k x_k^T] = E[(h_i x_k + \nu_k)x_k^T]$$

$$E[y_k(Y_k^-)^T R^{-T}C^T] = h_i CR^{-T}C^T e_1^T HR^{-T}C^T = h_i CR^{-T}C^T$$

$$e_1^T OCR^{-T}C^T = h_i CR^{-T}C^T$$

and it has been shown above that $CR^{-T}C^T$ is nonsingular, thus

$$e_1^T O = h_i = first\ row\ of\ O = h_c$$

The "only if" part of the proof is as follows: since the given model is an innovations model, the state must be given by $x_k=GY_k^-$ for some rank n matrix $G_{n\times\infty}$. $(F_i, h_i)=(F_c, h_c)$ implies that $H=OC$ by the factorization in (4.6). The state x_k is a basis for $\{Y_k^+\}|\{Y_k^-\}$, and by the projection theorem, $\{x_k\} = \{HR^{-1}Y_k^-\}$. (Recall the braces represent the space generated by the linear span of the enclosed random variables.) In order to obtain the state x_k, HR^{-1} (which is $\infty\times\infty$) must be replaced by the matrix G (which is $n\times\infty$), where G is defined by the requirement that it have the same row-space as HR^{-1} (so that $\{HR^{-1}Y_k^-\} = \{GY_k^-\}$) which is expressed as follows:

$$Range(G^T) = Range(R^{-T}H^T) = Range(R^{-T}C^T O^T) = Range(R^{-T}C^T) \tag{A.13}$$

where the last equality is true because O is of full rank. From (A.13), $G^T=R^{-T}C^T N^T$ or

$$G=NCR^{-1} \tag{A.14}$$

for some nonsingular matrix N. From the output equation of (4.9) and the definition of the innovations ν_k, we get

$$y_k|\{Y_k^-\} = h_i x_k = h_i GY_k^-$$

From the projection theorem,

$$E[(y_k - h_i GY_k^-)(Y_k^-)^T] = 0$$

$$e_1^T H = h_i GR$$

$$e_1^T OC = e_1^T OGR = e_1^T ONC \tag{A.15}$$

Let $w=O^T e_1$ and notice that $w \neq 0$. Then from (A.14)

$$C^T w = C^T N^T w \quad or \quad C^T(I-N^T)w = 0$$

Since C^T is of full rank and $w \neq 0$, $I=N^T$, and from (A.14), $G=CR^{-1}$.

REFERENCES

1. B.C. Moore, "Singular value analysis of linear systems, parts I,II," Dept. Elec. Eng., Univ. Toronto, Toronto, Ont., Syst. Contr. Rep 7801 and 7802, July 1978; also in *Proc. 17th IEEE Conf. Dec. Contr.*, pp. 66-73, 1979.
2. -----, "Principal component analysis in linear systems: controllability, observability, and model reduction," *IEEE Trans. Automat. Contr.*, vol. AC-26, pp. 17-31, 1981.
3. L. Pernebo and L.M. Silverman, "Balanced systems and model reduction," *IEEE Trans. Automat. Contr.*, vol. AC-27, pp. 382-387, 1982.
4. U.B. Desai and D. Pal, "Canonical variables, balanced stochastic realizations, and model reduction," Tech. Rep., Dept. Elec. Eng., Washington State University, Pullman, WA, 1982.
5. -----, "A realization approach to model reduction and balanced stochastic realizations," Proc. 16th Annual Conf. Info. Sci. Syst., Princeton University, Princeton, N.J., 1982.
6. V.C. Klema and A.J. Laub, "The singular value decomposition: its computation and some applications," *IEEE Trans. Automat. Contr.*, vol. AC-25, pp.164-176, 1980.
7. A.J. Laub, "On computing balancing transformations," *Proc. 1980 JACC*, FA8-E,1980.
8. I.C. Gohberg and M.G. Krein, *Introduction to the theory of linear nonselfadjoint operators*, Transl. Math. Monog., vol. 18, Amer. Math. Soc., 1969.
9. Y. Baram, "Realization and reduction of Markovian models from nonstationary data," *IEEE Trans. Automat. Contr.*, vol. AC-26, pp.1225-1231, 1981.
10. H. Akaike, "Markovian representation of stochastic processes by canonical variables," *SIAM J. Contr.*, vol. 13, pp. 162-173, 1975.
11. G.S. Sidhu and U.B. Desai, "New smoothing algorithms based on reversed-time lumped models," *IEEE Trans. Automat. Contr.*, vol. AC-21, pp.538-541, 1976.
12. L. Ljung and T. Kailath, "Backwards Markovian models for second-order stochastic processes," *IEEE Trans. Inform. Theory*, vol. IT-22, pp.488-491, 1976.
13. P. Faurre, "Stochastic Realization Algorithms," in *System Identification: Advances and Case Studies*, Ed. R.K. Mehra and D.G. Lainiotis, New York: Academic Press, 1976.
14. G. Verghese and T.Kailath, "A further note on backwards Markovian models," *IEEE Trans. Inform. Theory*, vol IT-25, pp.121-124, 1979.
15. T. Kailath, *Linear Systems*, Englewood Cliffs, New Jersey: Prentice-Hall, 1980.

AN ALGORITHM FOR SINGULAR VALUE DECOMPOSITION AND ITS APPLICATION TO LINEAR SYSTEM MODEL REDUCTION

M.S. KAMEL and K. SINGHAL
Department of Systems Design
University of Waterloo
Waterloo, Ontario, Canada

An algorithm for determining the singular values of large low pseudo rank matrices is presented. It uses the implicit QR factorization technique [1] to generate a matrix with the same rank as the pseudo rank of the original matrix and having the same singular values. The usual singular value decomposition is then performed on this low rank matrix instead of the original matrix. This leads to substantial reduction in computation costs. The algorithm is particularly attractive for applications where the pseudo rank of the matrix is small relative to its size such as linear system model reduction in control and power systems [2,3].

The singular value decomposition consists of two distinct parts. In the first part the matrix is reduced to bidiagonal form while the second iteratively computes the singular values. As the major effort is in reducing the matrix to bidiagonal form, we could instead first transform it to upper trapezoidal form using the truncated QR factorization with column interchange and then compute the bidiagonal form. This strategy reduces CPU when the rank of the matrix is small compared to its size.

An even more efficient algorithm is obtained if we perform an implicit QR factorization and generate the trapezoidal matrix a row/column at a time [1]. This avoids operations on those parts of the matrix which will ultimately be discarded.

The bulk of the work done by the usual singular value decomposition algorithm is in the reduction to bidiagonal form. We compare the algorithms up to this step; the work done by the rest of the steps is comparable.

Consider determining the singular value decomposition of a matrix A of size nxn and a pseudo rank k, then:

(i) For SVD on A the bidiagonal step costs $\frac{4}{3} n^3$ flops.

(ii) For truncated QR, the factorization step costs $\frac{2}{3}[n^3 - (n - k)^3]$ flops.

(iii) The implicit QR factorization step costs $(kn^2 + 2k^2 n)$ flops.

In (ii) and (iii) the bidiagonal step costs $2(k^2 n - \frac{k^3}{3})$ flops and a cost of $\frac{2}{3} k^3$ flops is needed to transform back the right and left singular vectors. If we take k to be n/10 (in many practical applications k will be much less than that) the costs will be

SVD on A $\frac{4}{3} n^3$

Truncated-QR $\simeq (.17) \frac{4}{3} n^3$ i.e. 83% savings

Implicit-QR $\simeq (.024) \frac{4}{3} n^3$ i.e. 97% savings

The algorithm has been implemented and tested on a number of problems. The results reflect the savings as indicated above. We also studied the sensitivity of the computed singular values to the value of k. We found that one can obtain the large singular values to reasonable accuracy by stopping the factorization at earlier stages and work with matrices of smaller dimension.

References

[1] M.S. Kamel and K. Singhal, "An efficient algorithm for the factorization of possibly rank deficient matrices", 1982 Sparse Matrix Symposium, Fairfield Glade, Tennessee, Oct. 1982.

[2] S. Kung, "A new identification and model reduction algorithm via singular value decomposition", in Proc. of 12th Asilmar Conf. Circuit, Syst. and Comput., Pacific Grove, California, Nov. 1978, pp. 705-714.

[3] Y-K. Tsai, N. Narasimhamurthi, and F.F. Wu, "Structure-Preserving model reduction with application to power system dynamic equivalents", IEEE Trans. Circuits Syst., Vol. CAS-29, August 1982, pp. 525-535.

[4] C.L. Lawson and R.J. Hanson, Solving Least Squares Problems, Englewood Cliffs, N.J.: Prentice-Hall, 1974.

DISCRETE-TIME SYSTEMS WITH SLOW AND FAST MODES.

MINH T. TRAN, The Boeing Company, Wichita, Kansas
MAHMOUD E. SAWAN, Wichita State University, Wichita, Kansas

INTRODUCTION

Methods of design and analysis of continuous linear singularly perturbed systems have been well documented [1, 2]. However, in a recent survey [3], it was noted that singular perturbation results for dynamical systems defined in discrete-time are not readily available and that a number of potential applications suggest a valuable role for such results. In this paper, a class of large scale discrete-time systems which contain slow and fast modes is considered. A method is proposed for a complete separation of slow and fast subsystems which will approximate the original system.

SUMMARY

As shown in [4], a discrete-time system which satisfies the two-time-scale property can be described as

$$X(n+1) = A_{11}X(n) + A_{12}Z(n) + B_1U(n), \quad X(0) = X_0 \tag{1a}$$

$$\alpha[Z(n+1) - Z(n)] = A_{21}X(n) + (A_{22} - I)Z(n) + B_2U(n), \; Z(0) = Z_0 \tag{1b}$$

$$Y(n) = C_1X(n) + C_2Z(n) \tag{1c}$$

where $X \varepsilon R^{n_1}$, $Z \varepsilon R^{n_2}$, $U \varepsilon R^{m}$, $Y \varepsilon R^{k}$, A_{11}, A_{12}, A_{21}, A_{22}; B_1, B_2, C_1, C_2 are matrices of appropriate dimensions and α is a small positive scalar. As indicated in [4], the fast modes corresponding to eigenvalues clustering around the origin of the unit circle can be neglected by setting $\alpha = 0$ resulting in the following system (with the bar indicated $\alpha = 0$)

$$\overline{X}(n+1) = A_{11}\overline{X}(n) + A_{12}\overline{Z}(n) + B_1\overline{U}(n), \quad \overline{X}(0) = X_0 \tag{2}$$

$$0 = A_{21}\overline{X}(n) + (A_{22} - I)\overline{Z}(n) + B_2\overline{U}(n) \tag{3}$$

$$\overline{Y}(n) = C_1\overline{X}(n) + C_2\overline{Z}(n) \tag{4}$$

Assuming that $(I - A_{22})$ is nonsingular, $Z(n)$ can be obtained from (3) and substituted back in (2) and (4), the slow subsystem of (1) can be described as

$$X_s(n+1) = A_0X_s(n) + B_0U_s(n), \quad X_s(0) = X_0 \tag{5a}$$

$$Y_s(n) = C_0X_s(n) + D_0U_s(n) \tag{5b}$$

where $A_0 = A_{11} + A_{12}(I - A_{22})^{-1}A_{21}, \quad B_0 = B_1 + A_{12}(I - A_{22})^{-1}B_2$ (6a)

$$C_0 = C_1 + C_2(I - A_{22})^{-1}A_{21}, \quad D_0 = C_2(I - A_{22})^{-1}B_2 \tag{6b}$$

The fast subsystem can also be derived by assuming that the slow variables are constant during the fast transients. Notice that

$$\alpha[\overline{Z}(n+1) - \overline{Z}(n)] = A_{21}\overline{X}(n) + (A_{22} - I)\overline{Z}(n) + B_2\overline{U}(n) \tag{7}$$

and along with (1b), we can write

$$\begin{aligned}\alpha[Z(n+1) - Z(n)] = &-(A_{22} - I)\overline{Z}(n) - B_2\overline{U}(n) + \alpha[\overline{Z}(n+1) - \overline{Z}(n)] \\ &+ (A_{22} - I)Z(n) + B_2U(n)\end{aligned} \tag{8}$$

or

$$\begin{aligned}\alpha[\{Z(n+1) - \overline{Z}(n+1)\} - \{Z(n) - \overline{Z}(n)\}] = &(A_{22} - I)\{Z(n) - \overline{Z}(n)\} \\ &+ B_2\{U(n) - \overline{U}(n)\}\end{aligned} \tag{9}$$

By letting $Z_f(n) = Z(n) - \overline{Z}(n)$, $U_f(n) = U(n) - \overline{U}(n)$, $Y_f(n) = Y(n) - \overline{Y}(n)$, the fast subsystem can be obtained as

$$\alpha[Z_f(n + 1) - Z_f(n)] = (A_{22} - I)Z_f(n) + B_2U_f(n), \quad Z_f(0) = Z_0 - \overline{Z}(0) \tag{10a}$$

$$Y_f(n) = C_2Z_f(n) \tag{10b}$$

Suppose now the subsystem controllers which are to be designed according to the slow and fast mode performance specifications, have the form

$$U_s(n) = G_sX_s(n) \quad \text{and} \quad U_f(n) = G_fZ_f(n) \tag{11}$$

then along with $\overline{X}(n) = X_s(n)$, $\overline{U}(n) = U_s(n)$, (3) can be rewritten as

$$\overline{Z}(n) = (I - A_{22})^{-1}[A_{21} + B_2G_s]X_s(n) \tag{12}$$

and we can define the composite control as

$$\begin{aligned} U_s(n) + U_f(n) &= G_sX_s(n) + G_fZ_f(n) \\ &= [\{I + G_f(A_{22} - I)^{-1}B_2\}G_s + G_f(A_{22} - I)^{-1}A_{21}]X_s(n) \\ &\quad + G_f[-(A_{22} - I)^{-1}(A_{21} + B_2G_s)X_s(n) + Z_f(n)] \end{aligned} \tag{13}$$

With $X(n)$ replacing $X_s(n)$ and $Z(n)$ replacing $\overline{Z}(n) + Z_f(n)$, we have the following lemma

Lemma If the controls

$$U_s(n) = G_sX_s(n) \tag{14}$$

$$U_f(n) = G_fZ_f(n) \tag{15}$$

$$U(n) = [\{I + G_f(A_{22} - I)^{-1}B_2\}G_s + G_f(A_{22} - I)^{-1}A_{21}]X(n) + G_fZ(n) \tag{16}$$

are applied to systems (5), (10) and (1), respectively, and if $(A_{22} - I + B_2G_2)$ is stable, then

$$X(n) = X_s(n) + O(\alpha)$$

$$Z(n) = \overline{Z}(n) + Z_f(n) + O(\alpha)$$

$$U(n) = U_s(n) + U_f(n) + O(\alpha)$$

$$Y(n) = Y_s(n) + Y_f(n) + O(\alpha)$$

hold for all $n \geq 0$.

Due to limited space, the proof of this lemma is omitted from this summary. This lemma suggests that subcontrollers be separately designed to satisfy the corresponding mode performance criteria and implemented as the composite controller as in (16). Development of separated designs for discrete optimal linear regulator problem is being investigated by the authors and will be the subject of another paper.

REFERENCES

[1] J. H. Chow and P. V. Kokotovic, "A Decomposition of Near Optimum Regulators for Systems with Slow and Fast Modes," IEEE Trans. Automat. Contr., Oct. 1976.

[2] M. T. Tran and M. E. Sawan, "Singularly Perturbed Systems with Low Sensitivity to Model Reduction", to appear in The Journal of Astronautical Sciences.

[3] N. Sandel et. al., "Survey of Decentralized Control Methods for Large Scale Systems," IEEE Trans. Automat. Contr., 1978.

[4] M. T. Tran and M. E. Sawan, "On the Robustness of Reduced-Order Discrete-Time Models," Proc. of 1st ACC Conference, June 1982.

ON THE OPTIMAL PLACEMENT OF RECOVERY POINTS

MOON-JUNG CHUNG and M. S. KRISHNAMOORTHY
Rensselaer Polytechnic Institute
Troy, New York 12181

ABSTRACT

In this paper, we study the complexity of placing recovery points in computer programs, so that the roll back time is minimized. The roll back time corresponds to the recomputation time involved from the previous correct point. In this paper we give polynomial time $(O(n^2))$ algorithms for placing recovery points, when the underlying program model is either a path tree or a rooted tree. We also show that the problem is NP-complete, when the underlying program model is a directed graph. We also show an approximation algorithm which is within factor 2 of optimal solution for distributed systems.

1. INTRODUCTION

We use a directed graph representation of computer programs. The nodes in the digraph represent events, and the edges denote the computation. The weight on an edge indicates the time required to perform the computation.

The recovery points are used to store the computation performed upto certain points of computation. We assume that the recovery points are placed on some subset of nodes of the digraph. In our model, we assume that the time required to store and retrieve the partial computation is constant and equal over all partial computations. The save time and load time can be easily incorporated in our model, and our results may be generalized.

The time taken between two successive recovery points is the sum of weights on the edges of the path between these nodes where the recovery points are placed. If there is more than one path between these two recovery points, we calculate the maximum of the time among all paths. This is also called the roll-back time (1,2). Intuitively, the roll-back time is the time to perform the recomputation, assuming the answers are correct on the previous recovery point and wrong at the current recovery point.

There are many performance measures one could consider on the roll-back time. We will consider the measure which takes the maximum among all roll-back times.

Formally this is defined as follows:

We are given a digraph $G(V,E)$, where V is a set of nodes, and E is a set of directed edges and there is a weight function $W : E \rightarrow R$. We will assume $|V| = n$, and $|E| = m$. We are required to place k recovery points on the nodes of G, such that the roll-back time is minimized i.e., find a subset $V_1 \subseteq V$, $|V_1| = k$, such that

$$\max_{1 \leq l \leq k} \max_{\substack{\text{recovery points are placed}\\ \text{over } v_i, v_j \text{ and no other}\\ \text{recovery points are placed}\\ \text{between them.}}} (d(v_i, v_j))$$

is minimized. This problem is referred to as optimal placement of recovery points. For example, consider the Fig. 1.1 where we are allowed to place only $k = 2$ recovery points. The maximum roll-back time is minimized, if we place the recovery points on nodes v_2 and v_3. The minimum roll-back time is 11.

In all cases, we assume a fictitious recovery point placed at the source (start) and sink (stop) nodes. It is reasonable to assume that there will always be a source and sink node. This model could easily be generalized for concurrent programs or distributed systems.

2. POLYNOMIAL TIME ALGORITHM ON TREES

In this section we describe $O(n^2)$ algorithms for finding the minimum roll-back time when the underlying digraph is a rooted tree. Thus if underlying digraph is a simple path we can solve $O(n^2)$ time.

To solve the problem consider the following decision problem.

Input: A rooted tree $T(V,A,r)$, k, and R.

Question: Is it possible to place k recovery points on some nodes of the digraph, such that the roll-back time never exceeds R.

First we will give algorithms to solve this decision problem and then solve the optimal placement problem on trees.

As usual we will assume that there are fictitious recovery points placed on the root node and on leaf nodes.

Let leafnode (T) be the set of all leaf nodes of tree T and parent(v) be the parent node of v.

Algorithms for the decision problem on trees.

Input: A rooted tree T, positive integer R, and k.

Output: A placement of k recovery points on T with roll-back time at most R if there is such placement, else answer "no".

Algorithm:

```
   begin
   q:=0;sum:=0;flag:=true;
   mark(u) :=false for all node u in T;
   while flag do
     for each v in leafnode (T) do
     begin
         while parent (v) <> root and  mark(parent(v)) do
         begin
            u:=parent (v);sum:=sum+W(u,v);
            if W(u,v) > R then begin
                             write  (' no');flag:=false;
                           end;
            if sum > R then begin
                             q:=q+1;mark(v):=true;sum:=0;
                       end;
            if q > k then flag:=false; write('no');
            v:=u;
         end; (*inner while  *)
    end; (* outer while *)
  end;
```

We can prove that the algorithm correctly solves the decision problem by contradiction.

The Boolean variables mark(v) associated with nodes are used to avoid the recomputations. The algorithm hence takes linear time.

Once the recognition problem has been solved, the optimization problem can be solved by calling the decision problem at most n times. To show this we need the following lemma. [6]

Lemma 1: An integer x between 1 and B can be determined by logB questions of the form "Is $x > a$"?

The roll-back time with optimal placement of recovery points is less than the total computation time Tmax and $\text{Tmax} < 2^n$, where n is the input length of the problem. Thus we need $(\log \text{Tmax}) \leq n$ questions for the decision problem. Since each decision problem takes at most N step (where N is the n^2 number of nodes of T and $N < n$), it takes at most $N \cdot n < n^2$ time to find the optimal placement of recovery points for rooted tree T.

3. NP-COMPLETENESS

We will prove that the optimal placement of recovery points becomes NP-complete, even if we have acyclic digraphs.

Consider the following decision problem.

Optimal Placement of Recovery Point Problem

Input: A directed acyclic graph $G = (V,E)$, k, and R.

Question: Is it possible to place k recovery points on some nodes of the digraph, such that the roll-back time never exceeds R.

Lemma 2: The 3-SAT problem is polynomial time transformable to the optimal placement of recovery points problem.

Proof: Let $W = C_1 C_2 \ldots C_n$ be a well formed formula over variables $x_1, x_2, \ldots, x_n$. From W we will construct a weighted directed acyclic graph G(W) as follows.

We will construct a structure for each of the variable x_i as given in Fig. 3.1 and construct a structure for each clause C_j as shown in Fig. 3.2. If literal x_i ($\bar{x}_i$) appears as the k^{th} term in the j^{th} clause connect node x_i ($\bar{x}_i$) to $\langle C_j', k \rangle$. For example the graph $G(W_1)$ can be constructed from a wff $W_1 = (x_1 + x_2 + x_3)(\bar{x}_1 + \bar{x}_2 + x_3)(x_1 + \bar{x}_2 + \bar{x}_3)$.

Now let us prove that we can place $k = 3r + n$ recovery points such that roll back time is at most 4 iff W_1 is satisfiable.

(if) Let g be an assignment which makes W true. We will put recovery points as follows. If x_i ($\bar{x}_i$) is assigned true under g put a recovery point on x_i ($\bar{x}_i$). If k^{th} term in j^{th} clause is true under g then put a recovery point on $<C_{j,}k>$ and if k^{th} term in j^{th} clause is false under g then put a recovery point on $<C'_{j,}k>$. Thus there are 3r + n recovery points in G. For each path $a_i - x_i - \bar{x}_i - b_i$ one node has a recovery point. For each path $d_j - <C_{j,}1> - <C_{j,}2> - <C_{j,}3> - e_j$, at least one node has a recovery point since W is satisfiable. Moreover there is no path from the structure of x_i to the structure C_j with weight more than 4 without passing through a node having recovery point. Thus maximum roll back time of G is at most 4 by the above strategy.

(only if) Suppose that G(W) has roll back time at most 4 with 3r + n recovery points. For each path $a_i - x_i - \bar{x}_i - b_i$, one node has a recovery point. Let x_i be connected to $<C_{j,}k>$. Then either $<C_{j,}k>$ or $<C'_{j,}k>$ must have a recovery point. Since each clause structure can have at most 3 recovery points, only one of $<C'_j,k>$ or $<C_j,k>$ can have a recovery point. Thus if x_i does not have a recovery point then $<C'_j,k>$ must have a recovery point. (If not then the path $a_i - x_i - <C'_j,k> - <C_j,k>$ has weight 6.) By the same reason, if $\bar{x}_i$ appears as the k^{th} term in C_j and x_i does not have a recovery point, then $<C'_j,k>$ must have a recovery point. Consider an assignment g which assigns x_i to be true if x_i has a recovery point and $\bar{x}_i$ true if $\bar{x}_i$ has a recovery point. Such an assignment g is consistent. Since for each j, at least one of $<C_j,1>$, $<C_j,2>$ or $<C_j,3>$ has a recovery point, such an assignment makes W true.

For the example the recovery points may be placed on x_1, $\bar{x}_2$, x_3, $<C_1,1>$, $<C'_2,1>$, $<C_3,1>$, $<C'_1,2>$, $<C_2,2>$, $<C_3,2>$, $<C_1,3>$, $<C_2,3>$, $<C'_3,3>$ of $G(W_1)$ in Fig. 3.3 and such placing corresponds to the assignment of x_1 = true, x_2 = false, x_3 = true.

Thus we have proved that it is possible to place 3r + k recovery points with roll back time 4 if and only if the given 3-SAT problem W is satisfiable.

It is clear that the optimal placement problem is in NP. Hence we have the following theorem.

Theorem: The optimal placement problem is NP-complete.

4. DISTRIBUTED MODELS

In a distributed system we need to consider the interprocess communications. An error in one process can be propagated to another process through communication. This will directly affect the technique of implementing recovery points. If an error is detected in one process then the process roll back to the most recent recovery point. It nullifies all communications the process made with other processes between recovery points and error detection points, and this nullification of communication may cause other processes to roll back. It may cause a roll back propagation, called domino effect. Fig. 4 illustrates the domino effect.

We assume that a distributed system consists of several straight line programs which communicate at various points. We also assume that the model is static, i.e., we know the points when communications occur. A communication graph is the graph which shows all communications between processes from the start to the end. In this paper we assume that a communication graph is given and we want to minimize the roll back time under this model. We will use <t,i> to denote the point at time t of process i. Now let us state the problem formally.

Recovery Point at Distributed System.

Input: A communication graph of p processes from time 0 to Tmax, an integer K which denotes the number of recovery points, and a set V of tuples <t,i>, where if <t,i> is in V then we can place a recovery point at time t for process i.

Output: Find a subset V' of V such that $|V'| = K$ and if we place recovery points at each point in V' then the roll back time is minimized.

Let us assume that we put recovery points at the nodes in V'. We say that there is a path from <t,i> to <t',j> if $t<t'$ and there is a path connecting from <t,i> to <t',j> such that the edges in the path are either communication line or directed to downward. Let us also assume that we can put a recovery point at any point just before each communication.

We will describe a approximation time algorithm which is optimal within factor 2. That is the roll back time of G by approximation algorithm is at most 2 times the rollback time of optimal placement with K recovery points.

Let C = ($<t_1,1>, <t_2,2>, \ldots, <t_p,p>$) be a set of points of G where we can put recovery points. We say C is a cut if, by removing the points in C, G is divided into two connected components. Or C is a cut iff the following condition is true: for all i and j if $t<t_i<t_j<t'$ then there is no path from $<t,i>$ to $<t',j>$ without passing through any node in C. Note that each processor has exactly one recovery point.

Lemma 3: Let C = ($<t_1,1>, <t_2,2>, \ldots, <t_p,p>$) and C' = ($<t'_1,1>, \ldots, <t'_p,p>$) be cuts such that $t_i < t'_i$. If we put recovery points on all nodes in C and C' then the maximum roll back time between C and C' is $d(C,C') = \max\limits_{i,j} |t'_1 - t_j|$.

Proof: Since C and C' are cuts, there is no path from $<t,i>$ to $<t',j>$ such that $t < t_i$ and $t' > t'_j$. Thus lemma holds. We will say d as the distance between C and C'.

Let R' be the maximum distance of two adjacent points in the same process where we can put recovery points.

Formally

$$R' = \max_i \{t-t' \mid t > t', <t,i> \text{ and } <t',i> \text{ is in V, and there is no } <t'',i> \text{ in V such that } t' < t'' < t\}$$

For example in Fig. 4.1 R' is 6 . The roll back time for G should be at least R'.

Approximation Algorithm for Recovery Points

```
Input:  Communication graph G of p processors from time 0 to T,
        the number of recovery points K(K is multiple of p).
Output: Placement of recovery points for G.
Algorithm:
 1. Compute $q = Tp/K$.  (q is the interval of cut)
 2. If $R' > q$ then set $q = T/R'$
 3. Set $m = T/q$  (each process has m recovery points)
 4. For i = 0 to m - 1 do
       Comment:find the first cut $C_i$ from time $i \cdot q$ to $(i+1) \cdot q$
 5.    $C_i = 0$
 6.    For j = 1 to p do
 7.       Find a smallest $t_j \geq iq$ such that at time $t_j$ we can put a
          recovery point at $<t_j,j>$.
 8.       Put $<t_j,j>$ into $C_i$.
 9. end
    Comment: Now $C_i$ is a cut.
10. Place recovery points on $C_i$.
11. end
```

Since $q < R'$ we can always find a point such t_j at line 7. We can show that in optimal placement of recovery point with K recovery points for P processors the roll back time is at least $q = \max(Tp/K, R')$.

Since any C_i forms a cut and the distance between two successive cut is at most 2q, the roll back time is at most 2q. Thus our algorithm is optimal within factor of 2.

4. REFERENCES

1. Chandy, K. M. and Ramamoorthy, C. V., "Roll back and recovery strategies for computer programs", IEEE Trans. on Comp., June 1972, pp. 546-556.
2. Daouk, N., "Optimal checkpointing in real-time multiprogramming systems", Ph.D. Dissertation, Dept. of Electrical Engineering, University of Southern California, June 1979.
3. Garey, M. R. and Johnson, D. S., Computer and Intractability, W. H. Freedman and Company, 1979.
4. Kim, K. H., "Approaches to mechanization of the conversation scheme based on monitors," IEEE Trans. on Software Engineering, vol. SE-8, No. 3, May 1982, pp. 189-197.
5. Kim, K. H., "An approach to programmer-transparent coordination of recovering parallel processes and its efficient implementation rules", Proc. 1978 Int'l Conf. on Parallel Processing, Aug. 1978, pp. 58-68.
6. Papadimitriou, C. H. and Steiglitz, K., Combinatorial Optimization, Prentice Hall, 1982, pp. 171.
7. Randell, B., "System structure for software tolerance", IEEE Trans. on Software Engr., June 1975, pp. 220-232.
8. Randell, B., Lee, P. A., and Trelaven, P. C., "Reliability issues in computing system design", Computing Surveys, June 1978, pp. 123-165.

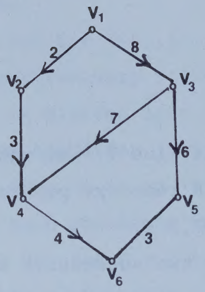

V1
2
8
V2
V3
7
3
6
V4
V5
4
3
V6

Fig. 1.1

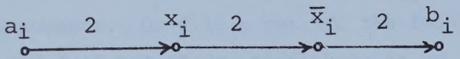

a_i
2
x_i
2
$\overline{x}_i$
2
b_i

Fig. 3.1

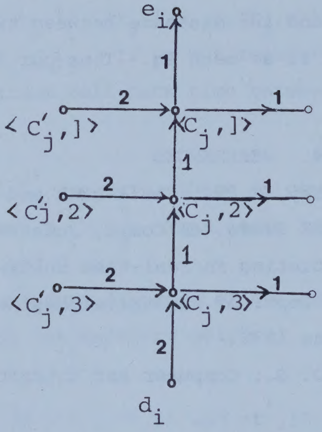

e_i
1
2
1
$\langle C'_j,1\rangle$
$\langle C_j,1\rangle$
1
2
1
$\langle C'_j,2\rangle$
$\langle C_j,2\rangle$
1
2
1
$\langle C'_j,3\rangle$
$\langle C_j,3\rangle$
2
d_i

Fig. 3.2

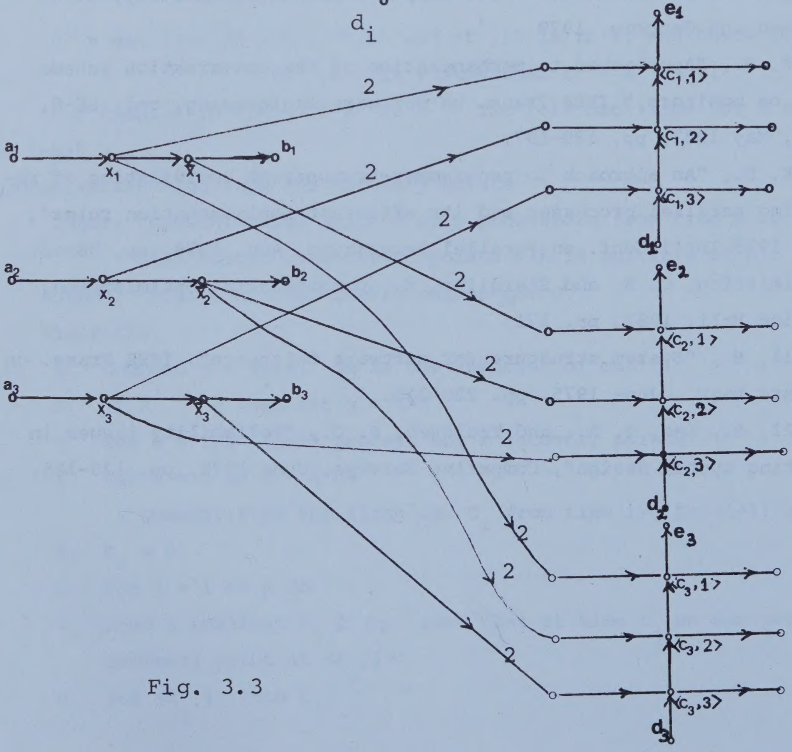

e_1
$\langle c_1,1\rangle$
$\langle c_1,2\rangle$
$\langle c_1,3\rangle$
d_1
a_1
x_1
$\overline{x}_1$
b_1
a_2
x_2
$\overline{x}_2$
b_2
a_3
x_3
$\overline{x}_3$
b_3
2
2
2
2
2
2
2
2
2
e_2
$\langle c_2,1\rangle$
$\langle c_2,2\rangle$
$\langle c_2,3\rangle$
d_2
e_3
$\langle c_3,1\rangle$
$\langle c_3,2\rangle$
$\langle c_3,3\rangle$
d_3

Fig. 3.3

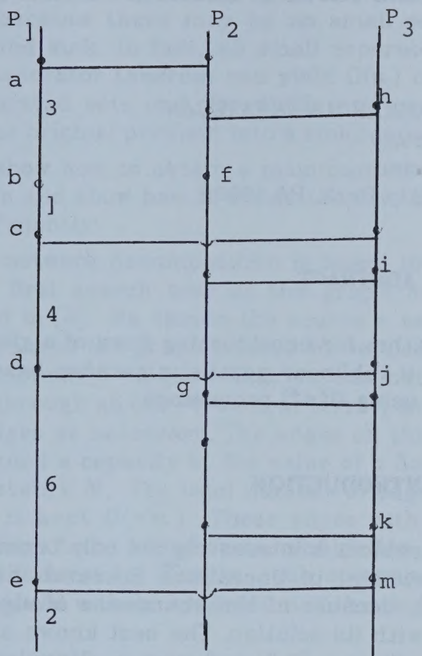

P]
P2
P3
a
3
b
]
c
4
d
6
e
2
f
g
h
i
j
k
m

Fig. 4.]

If we put recovery points at node a,b,c,e,f,g,h,i,j, k,m and if error is detected at m then all communications become nullified and process P], P2, and P3 must start from the beginning.

USING DIVIDE AND CONQUER TO FIND FLOWS IN DIRECTED PLANAR NETWORKS IN $O(n^{3/2}\log n)$ TIME[1]

DONALD B.JOHNSON and SHANKAR M.VENKATESAN
The Pennsylvania State University
Department of Computer Science
Whitmore Laboratory, University Park, PA 16802

ABSTRACT

We give an $O(n^{3/2}\log n)$ algorithm for constructing flows of a given value in planar directed networks and a sublinear parallel algorithm based on it, which runs in $O(\sqrt{n}\,(\log n)^3)$ time using $O(n^4)$ processors.

1. INTRODUCTION

The maximum network flow problem is interesting not only because of its importance as a fundamental problem in Operations Research but, more importantly in Computer Science, because of the abundance of algorithmic ideas that have been associated with its solution. The best known algorithm for the general problem is due to Sleator [7,8] and runs in $O(nm\log n)$ time on an n-vertex, m-edge network. This bound is $O(n^2\log n)$ on planar networks.

Our results are an $O(n^{3/2}\log n)$ algorithm for constructing flows of a given value in planar directed networks and a parallel algorithm based on it which runs in $O(\sqrt{n}\,(\log n)^3)$ time using $O(n^4)$ processors. The first of these results, together with the result due to Reif [5], yields an $O(n^{3/2}\log n)$ maximum flow algorithm in the case of planar undirected networks. The parallel bound leads to an $(\sqrt{n}\,(\log n)^3)$ parallel algorithm for finding a maximum flow in a directed planar network since the value of a maximum flow can be found in $O((\log n)^3)$ time using $O(n^4)$ processors [3]. The parallel algorithm is the first we know of with sublinear running time, the best former result being that of Shiloach and Vishkin [6], $O(n^2\log n)$ time using n processors.

We emphasize the $O(n^{3/2}\log n)$ result in the serial model because (1) it for the first time applies divide and conquer cleanly to a network flow problem, and (2) given the results in [3], the extension to a parallel algorithm is straightforward. The results are a significant departure from the traditional augmenting path or layered network approaches which do not seem to lend themselves to a divide and conquer solution nor to parallelize into efficient sublinear algorithms because the number of layers could be $\Omega(n)$.

It is natural to think of the planar separator theorem [4] as a starting point for divide-and-conquer on a planar graph (and we do indeed employ the basic lemmas of that theorem involving the properties of a breadth-first spanning tree in a planar graph). However, the planar separator itself is of no use because there may be no small separator that also separates the source and sink. In fact, all small separators could contain both. Also, the planar separator theorem can yield $\Omega(n)$ connected components in each of the separated sets and, for divide and conquer to work, it is desirable to divide the original problem into a small number of subproblems.

---

[1] This research partially supported by the National Science Foundation under grant MCS-8002684.

We show how to obtain a maximum of two subproblems at each level of recursion and show how to divide them so that their solutions can be recombined efficiently.

The network decomposition is based initially on the levels l_0 and l_1 in a breadth first search tree of the graph underlying the given network, as identified in [4]. We choose the source s as the root of the tree and identify components in the graph formed by vertices at level l_0 or higher. For each such component, we show that we can obtain a directed cycle that passes exactly through all the vertices at level l_0 in the component, embedding additional edges as necessary. The edges on this cycle for each such component are assigned a capacity v, the value of a flow that we are trying to construct in the network N. The total number of edges on the cycles for all such components is kept $O(\sqrt{n})$. These edges with capacity v allow flows from the subproblems to be added on recombination. A similar construction is carried out for the level l_1. Finally, each overcapacitated edge in the recombined flow is relaxed with an application of the technique of Itai and Shiloach [2].

2. PRELIMINARIES

Let $N=(\overline{G},c,s,t)$ be an n-vertex flow network, where $\overline{G}=(V,E)$ is a directed graph, $s,t \in V$ are the source and sink, and $c: V\times V \to \mathbf{R}^+$ is a capacity function. We require that $c(e)=0$ for $e \in V\times V-E$. An *$s-t$ flow* is a function $f: V\times V \to \mathbf{R}^+$ that satisfies conservation at each vertex except s and t and for which $0 \le f(e) \le c(e)$ for all pairs $e \in V\times V$.[2] The *value* of a flow is the net flow into t. A *pseudo flow* is a function that satisfies the conservation rule but not necessarily the capacity rule. Let $\overline{G}$ have some embedding in the plane. (Hereafter we let the symbol $\overline{G}$ stand for this *plane*, *i.e.* embedded, *graph*.)

Let $G=(V,E)$ be an n-vertex connected undirected plane graph derived in the usual way from $\overline{G}$. Let a breadth-first-search tree T in G be found in $O(n)$ time; T partitions V into levels $L_0, L_1, \cdots, L_{\max}$, where L_i is the set of vertices at distance i from the root of T. Let $L_{(\max+1)}$ be an extra empty level.

LEMMA 1 [Lipton and Tarjan, 4]. Two levels L_{l_0}, L_{l_1} that satisfy the following conditions can be found in $O(n)$ time:

(i) $0 \le l_0 < l_1 \le \max+1$,

(ii) $\sum_{i=0}^{l_0-1} \left| L_i \right| \ , \ \sum_{i=l_1+1}^{\max+1} \left| L_i \right| \ \le \ \frac{2n}{3},$

---

[2] While we define f on $V\times V$, it of course is sufficient to compute it only on E.

(iii) $\left|L_{l_0}\right| + \left|L_{l_1}\right| + 2(l_1-l_0-1) \leq c_1\sqrt{n}$,

for some constant c_1 . ▪

Let $\overline{G}(l)$ be defined as the plane subgraph induced by vertices in $\overline{G}$ at levels less than or equal to l, and let $(l)\overline{G}$ be the plane graph induced similarly by vertices in $\overline{G}$ at level greater than or equal to l. If $\overline{G}'$ is a subgraph of $\overline{G}$, then its embedding is derived from $\overline{G}$.

3. COMPONENTS OF THE GRAPHS $(l_0)\overline{G}$ AND $(l_1)\overline{G}$

Let the breadth first tree T be rooted at s. Thus s is the unique vertex at level 0.

The graph $(l_0)\overline{G}$ consists of mutually ***vertex disjoint*** components each of which has vertices at level greater than or equal to l_0. Let $\mu_{l_0} \in (l_0)\overline{G}$ be such a component. The only vertices in μ_{l_0} that are connected to vertices that are not in μ_{l_0} are the vertices at level l_0.

LEMMA 2. There exists a face $F(\mu_{l_0})$ of μ_{l_0} on which lie all vertices at level l_0 in μ_{l_0}.

Proof : The root of the tree T must get embedded *inside* some face F in μ_{l_0}. All vertices on F in μ_{l_0} are at a level greater than or equal to l_0. Since the path in the breadth first tree from the root to any vertex in μ_{l_0} that is not on F must pass through some vertex on F, it follows that all vertices in μ_{l_0} that are not on F must be at a level greater than l_0. Then choose $F(\mu_{l_0})=F$. ▪

For each $\mu_{l_0} \in (l_0)\overline{G}$, do the following:

(i) Identify the face $F(\mu_{l_0})$ in $O(\left|\mu_{l_0}\right|)$ time.

(ii) If $F(\mu_{l_0})$ has more than one vertex at level l_0, traverse $F(\mu_{l_0})$ and construct a clockwise directed cycle $ENCL(\mu_{l_0})$ that goes through all the vertices at level l_0 on $F(\mu_{l_0})$ and through no other vertices, adding edges as necessary and embedding them in $\overline{G}$.

Call the resulting graph $\overline{G}_0$.

LEMMA 3. The cycle $ENCL(\mu_{l_0})$ can be embedded in $\overline{G}$ for each component $\mu_{l_0} \in (l_0)\overline{G}$.

Proof : By the arguments in Lemma 2, the vertices of $\overline{G}$ that get embedded *inside* $F(\mu_{l_0})$ are at levels less than l_0 and cannot be connected to vertices on $F(\mu_{l_0})$ that are at a level greater than l_0 because these vertices would then be at a level no higher than l_0, a contradiction. Hence an edge connecting any two subsequent vertices at level l_0 on a traversal of $F(\mu_{l_0})$ can be embedded in $\overline{G}$. ▪

LEMMA 4. Constructing $ENCL(\mu_{l_0})$ for all $\mu_{l_0} \in (l_0)\overline{G}$ takes $O(n \log n)$ time.

Proof: Finding $F(\mu_{l_0})$ in a component of $(l_0)\overline{G}$ and traversing it takes $O(|\mu_{l_0}|)$ time. Each edge between successive vertices at level l_0 is checked for existence. Each such checking can be done in $O(\log n)$ time with $O(n \log n)$ preprocessing time. There can be no more than $O(|\mu_{l_0}|)$ edges checked for each component and the components are vertex disjoint. Embedding a new edge takes $O(1)$ time. Hence, the total time taken over all components is $O(\log n \sum_{\text{all } \mu_{l_0}} |\mu_{l_0}|)$ which is $O(n \log n)$. ▪

LEMMA 5. The tree T is a breadth first tree in $\overline{G}_0$.

Proof: The edges newly embedded in $\overline{G}$ are always between two vertices at the same level. ▪

Define the new graph $\overline{G}_{l_0}$ to be the subgraph $\overline{G}_0(l_0)$. Without loss of generality, assume that the vertex t is at a level greater than l_1.

The graph $(l_1)\overline{G}_0$ consists of mutually vertex disjoint components each of which has vertices at a level no less than l_1. Since t is at a level greater than l_1, it must belong to one of these components. Call this component μ_{l_1}. A lemma similar to Lemma 2 holds for μ_{l_1} and the level l_1. Construct and embed the cycle $ENCL(\mu_{l_1})$ for μ_{l_1} as before, passing exactly through the vertices at level l_1 in μ_{l_1}. Again, it can be proven that lemmas similar to Lemmas 3, 4, and 5 hold in this case.

Note that in the case of $(l_0)\overline{G}$, we embedded enclosing cycles for all components, while in the case of $(l_1)\overline{G}_0$, we embed an enclosing cycle for only that component μ_{l_1} which contains t.

Define $\overline{G}_{l_1}$ as the graph gotten by adding the edges in $ENCL(\mu_{l_1})$ to the component μ_{l_1}.

Let t_0 and s_1 be the unique vertices at levels l_0 and l_1 respectively on the path in the tree from s to t. Augment the graph further with an edge $<u_i, u_{i+1}>$, $i = l_0, \cdots, l_1 - 1$, where $<u_{l_0}, \cdots, u_{l_1}>$ is the subpath, of the path in tree T from s to t, that joins t_0 and s_1. Call the resulting graph $\overline{G}_1$.

It is clear that $\overline{G}_{l_0}$ and $\overline{G}_{l_1}$ are plane subgraphs of $\overline{G}_1$.

4. DEFINING THE SUBPROBLEMS

Define a capacity function c' as follows:

(i) For all edges on $ENCL(\mu_{l_0})$ for $\mu_{l_0} \in (l_0)\overline{G}$, let $c'(e) = v$.
(ii) For all edges on $ENCL(\mu_{l_1})$ for μ_{l_1} containing t, let $c'(e) = v$.

(iii) For all edges e in $<u_{l_0}, \cdots, u_{l_1}>$, let $c'(e)=v$.

(iv) For all other edges e, let $c'(e)=c(e)$.

Let c_0 be c' restricted to $\overline{G}_{l_0}$ and let c_1 be c' restricted to the graph $\overline{G}_{l_1}$. Now, define $N_0=(\overline{G}_{l_0},c_0,s,t_0)$ and $N_1=(\overline{G}_{l_1},c_1,s_1,t)$.

LEMMA 6. If N admits a flow of value v, then the networks N_0 and N_1 each admit a flow of value v.

Proof: Without loss of generality, assume there is only one component μ_{l_0} in $(l_0)\overline{G}$. Then, exactly the vertices at level l_0 lie on the cycle $ENCL(\mu_{l_0})$. Let a flow f of value v in N be given. Since t is at a level greater than l_1, the net flow into vertices on $ENCL(\mu_{l_0})$ from vertices in $\overline{G}(l_0-1)$ equals v. Then, traverse $ENCL(\mu_{l_0})$ and put in sufficient flows in the edges of capacity v so as to satisfy conservation at all vertices on $ENCL(\mu_{l_0})$ except the unique vertex t_0, into which the net flow will be v. This directly constructs a flow of value v in N_0. Similarly, a flow of value v can be constructed in N_1. If there is more than one component in $(l_0)\overline{G}$, only one component μ_{l_0} contains t and the above flow construction procedure is carried out for that component. For all other components μ_{l_0}, net flow into vertices of μ_{l_0} from vertices in $\overline{G}(l_0-1)$ equals zero. Hence, $ENCL(\mu_{l_0})$ can be traversed and sufficient flows put in the edges so that conservation holds at all the vertices on $ENCL(\mu_{l_0})$ for such a component μ_{l_0}. Thus flows of value v can be constructed in N_0 and N_1, given a flow of value v in N. ▪

Solve the subproblems N_0 and N_1 recursively. By Lemma 1, $|N_0|,|N_1| \le \frac{2n}{3}+O(\sqrt{n})$.

5. COMBINING THE SOLUTIONS OF THE SUEPROBLEMS

Let f_0 and f_1 be the flows of value v found in N_0 and N_1 respectively. Now, define a flow f' in $N'=(\overline{G}_1,c',s,t)$ as follows:

$$f'(e) = \begin{cases} f_0(e) & e \text{ in } N_0 \\ f_1(e) & e \text{ in } N_1 \\ v & e \text{ in } <u_{l_0}, \cdots, u_{l_1}> \\ 0 & \text{otherwise .} \end{cases}$$

Note that f' satisfies conservation at vertices t_0 and s_1, because of the flow pushed along the path $<u_{l_0}, \cdots, u_{l_1}>$ from t_0 to s_1. The function f is a flow in N'. For all the edges newly added to $\overline{G}$, (that is, the edges in $\{\overline{G}_1-\overline{G}\}$), define $c(e)=0$.

LEMMA 7 [Itai and Shiloach, 2]. Let a planar network N admit a flow of value v. Let f' be a pseudoflow of value v in N such that $f'(e)$ exceeds capacity in exactly one edge e. Then f' can be reduced to a flow f of value v in N in $O(n \log n)$ time. ▪

THEOREM 1. Let a planar network N admit a flow of value v. Then, the flow f' obtained from f_0 and f_1 can be reduced to a flow f of value v in N in $O(n^{3/2} \log n)$ time.

Proof: For each edge e in N' for which $f'(e)$ is greater than $c(e)$, relax the excess $f'(e)-c(e)$ as in Lemma 7 and set $c'(e)=c(e)$. If this operation fails on some overflowed edge, then it follows by induction on Lemma 7 that N does not admit a flow of value v. Otherwise, all overflowed edges will be relaxed, yielding a flow of value v in the planar network N (since all the newly added edges with zero capacity can now be removed).

Since the overflowed edges are either on the enclosing cycle for some component μ_{l_0} or μ_{l_1} or on the path $<u_{l_0}, \cdots, u_{l_1}>$, they cannot exceed $|L_{l_0}|+|L_{l_1}|+(l_1-l_0)$ in number, which, by Lemma 1, is $O(\sqrt{n})$. Hence, the time required to reduce the flow f' to a flow f of value v in N is $O(n^{3/2} \log n)$. ▪

Note that, if the original network N does not admit a flow of value v, our procedure will fail in one of two ways. Either one of the subproblems fails or the pseudoflow f' cannot be relaxed for some edge e. If the sink t is at a level no greater than l_1, then the network N_1 can be eliminated. If t is at a level i and i is $O(\sqrt{n})$, then both N_0 and N_1 are not needed.

6. RUNNING TIME

The construction of N_0 and N_1 can be done in $O(n \log n)$ time. Obtaining f from f_0 and f_1 takes $O(\sqrt{n} \cdot n \log n)$ time. Thus we have the recurrence for the running time $T(n)$:

$$T(n) \le C_1 \quad \text{if } n \le n_3$$

$$T(n) \le T(n_0) + T(n_1) + C_2\sqrt{n} \cdot n \log n \quad \text{if } n > n_3$$

where the following conditions hold:

n_3, C_1, C_2 are fixed ,

$n_0, n_1 \le 2n/3 + O(\sqrt{n})$, and

$n_0 + n_1 \le n + O(\sqrt{n})$.

The solution to this recurrence is

$$T(n) = O(n^{3/2} \log n).$$

7. MAXIMUM FLOW ALGORITHMS

The value of a minimum cut in a planar undirected network can be found in $O(n(\log n)^2)$ time [5]. So Theorem 1 implies the following:

COROLLARY 1. A maximum flow in a planar undirected network can be found in $O(n^{3/2}\log n)$ time. ▪

We also claim that the flow construction algorithm implied in Theorem 1 can be easily parallelized to run in $O(\sqrt{n}(\log n)^3)$ time, using a parallel version of the algorithm due to Hassin [1] to relax overflowed edges. Together with the results in [3], this implies the following:

COROLLARY 2. A maximum flow in a general planar network can be found in $O(\sqrt{n}(\log n)^3)$ parallel time using $O(n^4)$ processors. ▪

8. ACKNOWLEDGEMENT

The authors have recently obtained an $O(n\log n)$ flow construction algorithm for general planar networks. This will appear elsewhere.

9. REFERENCES

[1] Hassin, R., Maximum flow in (s,t)-planar networks, *Inf. Proc. Letters* **13**, 3(Dec. 1981) 107.

[2] Itai, A. and Y. Shiloach, Maximum flow in planar networks, *SIAM J. Comput.* **8**, 2(May 1979) 135-150.

[3] Johnson, D. B. and S. M. Venkatesan, *Parallel Algorithms for Minimum Cuts and Maximum Flows in Planar Networks*, Memorandum No. UCB/ERL M82/12, Univ. of California, Berkeley, February, 1982 (an extended version is to appear in *Proc. 23rd Annual Symp. on Foundations of Comp. Sci*).

[4] Lipton, R. J. and R. E. Tarjan, A separator theorem for planar graphs, *Proc. A Conference on Theoretical Computer Science*, University of Waterloo, August 1977.

[5] Reif, J., Minimum $(s-t)$-cut of a planar undirected network in $O(n\log^2 n)$ time, *Proc. Eighth Int. Colloq. on Automata, Languages, and Programming*, July 1981, Acre, Israel, (published as *Lecture Notes on Computer Science*, 115, Springer-Verlag, 1981) 56-67.

[6] Shiloach, Y., and U. Vishkin, *An efficient parallel max-flow algorithm*, Technical Report #196, The Technion, Haifa, Israel, February, 1981.

[7] Sleator, D. D., *An $O(nm\log n)$ Algorithm for Maximum Network Flow*, Ph.D. Dissertation, Computer Science Department, Stanford University (1980).

[8] Sleator, D. D., and R. E. Tarjan, A data structure for dynamic trees, *Proc. Thirteenth ACM Symp. on Th. Comp. Sci.*, May, 1981, 114-122.

SOME ALGORITHMS FOR SCHEDULING INDEPENDENT VECTOR TASKS ON A MULTIPIPE VECTOR PROCESSOR

SUDARSHAN K. DHALL and ALI DALILI-YAZDI
School of Electrical Engineering and Computer Science
University of Oklahoma, Norman, Oklahoma

ABSTRACT

The problem of scheduling a set of vector tasks on a vector processor with multiple identical pipelines is considered. The pipelines are assumed to be multifunctional, though only one type of function can be evaluated at any given time. The overhead due to start-up and flush time is taken into account in the model considered here. The set of vector tasks is assumed to be independent. Since the problem is known to be NP - complete, we consider heuristic algorithms. Two such algorithms are considered, whose time complexity is $O(m + n)$, where m is the number of pipelines and n is number of tasks. In each case, the bounds on the completion time of the schedules produced by the algorithms are also obtained.

1. INTRODUCTION

One of the motivating forces behind the development of the digital computer system was the need for quicker and more efficient ways to handle large numerical computations. The computational speed has increased manyfolds over the past three decades or so due to technological advancements. We, however, seem to have reached a stage where future increases in computational speed will be difficult to achieve merely by increasing the intrinsic speed of the logical components. There are many problems in weather prediction, nuclear physics, image processing etc., whose computational needs far exceed the capabilities of the computer systems with conventional Von Neumann architecture. Computer systems with other design techniques which exploit the inherent concurrency of operations in some problems have been in existence for quite some time. One such technique provides for vector operations. There are basically two types of vector processors-parallel vector processors and pipelined vector processors. An example of the parallel vector processor is the ILLIAC IV [1], in which an array of processing elements operates simultaneously on the elements of a vector. A pipelined vector processor provides for one or more arithmetic units each of which is pipelined. CDC-STAR-100 [2] and Texas Instruments ASC [3] are examples of this type of machines.

In a vector processor with multiple pipelines, the execution in parallel of concurrent vector tasks can lead to reduced execution time of a task system with proper and careful scheduling. In this paper, we consider the problem of scheduling a task system containing vector tasks on a multipipe vector processor. We assume that each pipeline in the processor is multifunctional in the sense that it is capable of evaluating different kinds of functions at different times, though at any given time only one function can be evaluated. All pipelines are assumed to be identical.

In general, the pipeline scheduling problem is NP-complete. Ramamoorthy and Li [4], and Bruno, Jones and So [5] studied the problem of scheduling scalar tasks on multiple pipelines, and obtained optimal scheduling algorithms for only restricted classes of problems. Li [6] obtained some optimal algorithms for (1) a task system with a tree-structured precedence relation in which all tasks require the same structure and the same number of initiations, and (2) for restricted vector loops.

The problem of scheduling vector tasks on multipipe vector processes was recently studied by Su and Hwang [7]. Their study included the system overhead cost involving startup and flush times for vector instructions.

They proved that the pipeline scheduling problem is NP-complete, and developed heuristic algorithms for special cases. In this paper, we give two algorithms for scheduling independent task systems and devise bounds on their performance. The worst-case performance of these algorithms is shown to be far better than that of the algorithm given in [7].

2. THE MODEL

We consider a multipipe vector processor containing m identical vector pipelines. Each pipeline is assumed to be multifunctional in the sense that it can be re-configured to perform different kinds of functions at different times. However, only one type of function can be evaluated at a time. The processor contains a vector controller which is responsible for scheduling the vector instructions on the available pipelines. Though the processor may be capable of performing scalar instructions also, we will be concerned with the scheduling of vector instructions only. The needed time for the completion of a vector instruction can be taken as:

$$t = t_o + t_\ell . L,$$

where

t_ℓ = latency time,

t_o = overhead due to start-up and flush time,

L = vector length (number of initiations).

The latency time, t_ℓ, is the time interval between two consecutive initiations. The start-up time is the time interval between the start of vector instruction and the initiation of the first operation, whereas the flush time is the time interval between the last initiation and the termination of the corresponding vector instruction [8]. We will define $t_\ell .L$ as the execution time of the vector instruction.

The task system to be scheduled can be specified by a quadruple $(\mathcal{T}, \mu, <, t_o)$, where $\mathcal{T} = \{T_1, T_2, \ldots, T_n\}$ is a set of n vector tasks, μ is a function from $\mathcal{T}$ to the set of real numbers R, with $\mu(T_i)$ indicating the execution time of the task T_i. There is also a partial ordering relation, <, defined on $\mathcal{T}$. $T_i < T_j$ implies that execution of T_j cannot begin until T_i has been executed. t_o is the overhead time of each vector task or a subtask of a vector task. The scheduling problem is to specify for each task T_i in the set as to the time interval(s) during which the task will be executed and the pipeline(s) that will be used to execute this task. Thus for each task T in $\mathcal{T}$, we specify a set of ordered pairs $\{([x_j, y_j], P_{i_j})\}_{j=1}^{k}$, $1 \leqslant i_j \leqslant m$, where an ordered pair ([x,y], P) means that pipeline P is to be used to process a subtask of T during the time interval [x,y], with the following constraints:

1. $\sum_{j=1}^{k} (y_j - x_j - t_o) = \mu(T)$ and $y_j - x_j - t_o > o$ for $j = 1,2,\ldots, k$.
2. If $i_{j_1} = i_{j_2}$, then $[x_{j_1}, y_{j_1}] \cap [x_{j_2}, y_{j_2}] = \Phi$, that is the two subtasks of a task cannot be processed by a pipe at the same time.
3. Let $S(T) = \min \{x_1, x_2, \ldots, x_k\}$ denote the starting time of task T, and $F(T) = \max \{y_1, y_2, \ldots y_k\}$ denote

the finishing time of T. If $T_i < T_j$, then $S(T_j) \geq F(T_i)$.

4. Different vector tasks cannot be processed by the same pipeline at the same time.

Any specification which satisfied all these requirements is called a schedule. The amount of time it takes to complete all the tasks in the task system is called the completion time of the task system. The following is an example of a task system and the specification of a schedule for the task system.

Example. Consider the task system $(T,\mu,<,t_o)$ with the precedent relation < depicted by the directed graph in Figure 1(a). Let $\mu(T_1) = 4$, $\mu(T_2) = 12$, $\mu(T_3) = 3$, $\mu(T_4) = 5$ and $\mu(T_5) = 6$. Let $t_o = 2$. The following is the specification of a schedule for this task system on three identical pipelines:

T_1: $\{([5,11], P_3)\}$, with $S(T_1) = 5$, $F(T_1) = 11$

T_2: $\{([0,8], P_1), ([0,8], P_2)\}$, with $S(T_2) = 0$, $F(T_2) = 8$

T_3: $\{([0,5], P_3)\}$, with $S(T_3) = 0$, $F(T_3) = 5$

T_4: $\{([8,15], P_1)\}$, with $S(T_4) = 8$, $F(T_4) = 15$

T_5: $\{([8,14], P_2)\ ([11,15], P_3)\}$, with $S(T_5) = 8$, $F(T_5) = 15$

In this schedule, T_2 is divided into two subtasks T_{21} and T_{22}, with execution time of 6 for each of them, and T_5 is divided into two subtasks, with $\mu(T_{51}) = 4$ and $\mu(T_{52}) = 2$. The completion time, ω, of the schedule is $\max\{F(T_1), F(T_2), F(T_3), F(T_4), F(T_5)\} = 15$. The schedule can be depicted diagramatically as in Figure 1(b). The shaded area represented the idle period of the processor.

For a given scheduling problem, we are interested in finding an optimal schedule, i.e. a schedule whose completion time is the minimum among all possible schedules. Our aim is to develop scheduling algorithms which will produce optimal schedules. A scheduling algorithm is merely a set of rules that determines a schedule for a given set of tasks.

In their paper [7], Su and Hwang have shown that in the special cases, when the partial ordering relation is empty or represents a tree, the following feasibility problem is NP-complete:

Given a vector task system, $(T,\mu,<,t_o)$, a vector computer with m identical pipelines, and a deadline D, does there exist a parallel schedule with completion time $\omega \leq D$?

It is, therefore, desirable to develop efficient algorithms which may not necessarily be optimal.

3. SCHEDULING INDEPENDENT VECTOR TASKS

In what follows, we confine ourselves to the case when the partial ordering relation is empty, i.e. all tasks are independent. Though our objective is to develop efficient scheduling algorithms which would produce optimal schedules in all cases, in view of the NP-completeness of the feasibility problem mentioned above, it seems very difficult to find such an algorithm. We will, therefore, attempt to develop efficient scheduling algorithms, which will produce schedules whose completion time is very close to the optimal completion time. (The algorithm is considered efficient, if its running time is some polynomial function of the number of inputs.) We have developed two such algorithms and have developed bounds on their worst-case performance.

Let the given task system contain n tasks. Let t_o be the overhead for scheduling each task. If a task is partitioned into subtasks, an overhead of t_0 will be required for scheduling each subtask. We define now $\omega_0 = \max \{t_0, \sum_{i=1}^{n} (\mu(T_i) + t_0)/m \}$. It is obvious that the completion time of any schedule has to be at least ω_0. In developing our algorithms, we try to keep the completion time as close to ω_0 as possible. The first algorithm is as follows.

ALGORITHM I:

```
1. /Initialize/
     $\omega = \omega_0$;            / $\omega_0$ as defined above/
   $i = 1$; $j = 1$; $s = 0$; $t = t + \mu(T_i)$;
2. While $i \leq n$ do
      begin: If $s + t_0/2 \geqq \omega$
                /extend estimated completion time and go to next
                 pipeline without scheduling the current task/
                then begin: If $s < \omega$ then     $\omega = \omega + t_0/2(m-j)$;
                                                     $j = j + 1$;
                                                     $s = 0$;

                     end
                else if $s + t \leqq \omega + t_0/2$
                     then begin: Assign remaining portion of $T_i$ to $P_j$
                                 over the time interval $[s, s+t]$;
                                 $s = s + t$;
                                 $i = i + 1$;
                                 $t = t + \mu(T_i)$;
                          end
                     else begin:
                          /task cannot be completed on the current
                           pipeline within the desirable limit, so
                           partition it into two parts, assigning one
                           part to the current pipeline/
                          Assign part of $T_i$ to $P_j$ over time interval
                          $[s, \omega + t_0/2]$;
                          $t = t - (\omega - s - t_0/2)$;
                          $\omega = \omega + t_0/2(m - j)$;
                          $s = 0$;
                          $j = j + 1$;
                          end;
      end;
```

The running time of this algorithm is O(n + m). Each task is first considered to evaluate ω. At the time of scheduling, the task is considered only once, if it can be scheduled on the pipe for which it is considered first. If it cannot be scheduled on the first pipe, it is considered again for scheduling on the next pipeline. Since there are only m pipelines, at most m - 1 tasks (or subtasks) can be considered twice. Thus the time complexity of the algorithm is O(n + m).

THEOREM 1: Let ω_1 be the completion time of the schedule produced by algorithm I on a processor with m identical pipelines. Let ω_o be the optimal completion time for a given task system. Then

$$\omega_o \leqq \omega_1 \leqq \omega_o + \left(\sum_{k=1}^{m-1} 1/k \right) t_o/2.$$

PROOF: As remarked above the minimum completion time is ω_o.

To establish the upper bound on ω_1, consider the following two cases:

Case 1: When no extension in ω is made.

In this case, each of the first m-1 pipelines must be busy for at least $\omega = \omega_o$ time units, but for not more than $\omega_o + t_o/2$ units. Moreover, no task should have been partitioned. Thus the total service time utilized on the first m-1 pipelines is at least $(m-1)\omega_o$. Since the total service time is only $m\omega_o$, the remaining time left for the m^{th} pipeline is less than or equal to ω_o. Hence all tasks finish on or before $\omega_o + t_o/2$ which is less than or equal to

$$\omega_o + \left(\sum_{k=1}^{m-1} 1/k \right) t_o/2.$$

Case 2: One or more extensions in ω are made.

There are two situations in which ω is extended. The first situation is that at the time scheduling of a task is considered, the pipeline has already been used for at least $\omega - t_o/2$ time units, but not more than ω time units. The other situation is when the last task scheduled on a pipeline is partitioned. In this case, the pipeline is used for $\omega + t_o/2$ time units. In either case, there is a loss of at most $t_o/2$ units of time for the intended use of that pipeline. In the first case, the pipeline is not used upto the intended time ω, but the usage falls short by at most $t_o/2$ units of time. In the second situation, the partition puts an additional requirement of t_o time units, whereas the pipeline has been overused for $t_o/2$ units. This loss of $t_o/2$ units of time is equally distributed over the remaining (m-j) pipelines, where j^{th} pipeline is the one which caused extension of ω, by extending ω to $\omega + t_o/2(m-j)$ for the remaining pipelines. Obviously, since the worst case would occur when there is an extension on every pipeline, the completion time must be less than or equal to

$$\omega_o + \left(\sum_{k=1}^{m-1} 1/k \right) t_o/2.$$

In the above algorithm, any loss of time made on the current pipeline is made up by extending the expected usage of the remaining pipelines. Thus the pipelines considered for usage earlier in the scheduling process may not be used as much as the remaining pipelines, particularly when there are numerous extensions of the expected usage time. In order to improve the completion time, it will help if we make a better estimate of the anticipated usage of the pipelines, which could be applied to all the pipelines. This is what we try to do in the following algorithm.

ALGORITHM II.

```
1. /Initialize/
     $\omega_2 = \omega_o + (m-1)t_o/m$;        /$\omega_o$ as defined above./
```

```
2. $i = 1$;  $j = 1$;  $s = 0$;  $t = t_o + \mu(T_i)$;
3. While $i \leq n$ do
       begin: If $s + t \leq \omega_2$
              then begin: Assign remaining part of $T_i$ to $P_j$
                          over the time interval $[s, s+t]$;
                          $s = s + t$;
                          $i = i + 1$;
                          $t = t_o + \mu(T_i)$;
                   end
              else if $s + t_o \geq \omega_2$
                        /task cannot be scheduled on this
                         pipeline. Consider the next pipeline./
                   then begin: $j = j + 1$;
                               $s = 0$;
                        end
                   else begin: Assign part of $T_i$ to $P_j$ over the
                               time interval $[s, \omega_2]$;
                               $s = 0$;
                               $t = t - (\omega_2 - s - t_o)$;
                               $j = j + 1$;
                        end
       end;
```

Since the "while .. do" loop in step 3 of the algorithm can be entered only for a maximum of n + m times, the time complexity of this algorithm is also O(n + m). The following theorem establishes the bounds for the completion time.

THEOREM 2. Let the completion time of the schedule produced by Algorithm II be ω. If ω_o is the optimal completion time, then

$$\omega_o \leq \omega \leq \omega_o + (m - 1)t_o/m.$$

PROOF: As before, the lower bound is obvious. So far as the upper bound is concerned, observe that only the last task on a pipeline is possibly partitioned. Thus the maximum number of possible partitions is $m - 1$. Each partition of a task increases the total execution time by t_o. On a pipeline on which there is no partition, the pipeline usage time is at least $\omega_2 - t_o$. Let the total number of partitions be k, where $0 \leq k \leq m - 1$. Then the total amount of time for which the first $m - 1$ pipelines are busy is greater than or equal to $k\omega_2 + (m - 1 - k)(\omega_2 - t_o)$. Since with k partitions, the total execution time is $m\omega_o + kt_o$, the remaining time for which the last pipeline need to be used is not more than $m\omega_2 - (m - 1)t_o + kt_o - k\omega_2 - (m - 1 - k)(\omega_2 - t_o) = \omega_2$. Since ω_2 is precisely the time we have earmarked for each pipeline, all the remaining tasks can be easily scheduled on the last pipeline. Thus the completion time is less than or equal to $\omega_2 = \omega_o + (m - 1)t_o/m$.

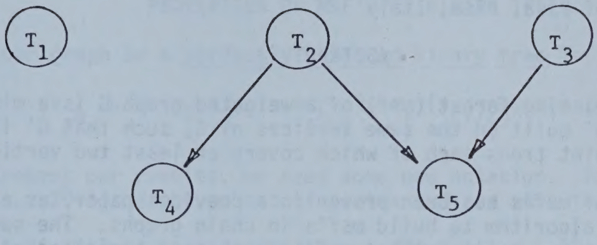
The upper bound on the completion time of the schedule produced by the algorithm given in [7] is $\omega_o + (m - 1)\, t_o/2$, and this bound is tight in the sense that examples can be easily constructed where this bound will actually be achieved. When $m > 2$, the upper bounds on the completion time of the algorithms considered in this paper are much lower.

4. CONCLUSIONS

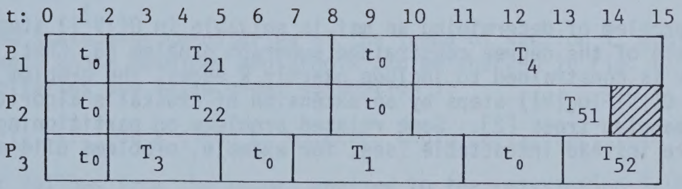
We have considered two simple algorithms for scheduling a vector task system on a vector processor with multiple identical pipelines. The vector tasks are assumed to be independent. For both the algorithms, bounds on the completion times are obtained. In both cases, the upper bounds are shown to be lower than the previously known such bounds.

REFERENCES

1. Barnes, G.H. et. al: "The ILLIAC IV Computer", IEEE Transactions on Computers, Vol. 17, 1968, pp. 746-757.
2. Hintz, R.G., and D. P. Tate: "Control Data STAR-100 Processor Design", Proc. COMPCON, IEEE No. 72CH 0659-3C, pp. 1-4.
3. Watson, W.J., "The Texas Instruments Advanced Scientific Computer", Proc. COMPCON, IEEE No. 72CH 0659-3C, pp. 292-293.
4. Ramamoorthy, C.V., and Li, H.F., "Sequencing Control in Multifunctional Pipeline Systems", Sagamore Computer Conference on Parallel Processing, 1975, pp. 79-89.
5. Bruno, J., J. W. Jones III, and K. So, "Deterministic Scheduling with Pipelined Processor", IEEE Transactions on Computers, April 1980, pp. 308-316.
6. Li, H.F., "Scheduling Trees in Parallel Pipelined Processing Environments", IEEE Transactions on Computers, Nov. 1977, pp. 1101-1112.
7. Su, S.P., and K. Hwang, "Multiple Pipeline Scheduling in Vector Supercomputers", Proceedings of the 1982 International Conference on Parallel Processing," August 1982, pp. 226-234.
8. Kogge, Peter M., "The Artchitecture of Pipelined Computers', McGraw-Hill Book Company, New York, New York, 1981.


T_1
T_2
T_3
T_4
T_5

(a) Graph describing the precedence relations in a task system.


t: 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
P_1
t_0
T_{21}
t_0
T_4
P_2
t_0
T_{22}
t_0
T_{51}
P_3
t_0
T_3
t_0
T_1
t_0
T_{52}

(b) Timing diagram for a schedule of the task system in (a)

FIGURE 1.

MINIMAL SPANNING FORESTS IN BINARY TREES[*]

FABRIZIO LUCCIO[**]
University of Hawaii, Honolulu, Hawaii

LINDA PAGLI
University of Pisa, Pisa, Italy

ABSTRACT

A minimal spanning forest (msf) of a weighted graph G is a minimum-weight subgraph G' built on the same vertices of G, such that G' is composed of disjoint trees each of which covers at least two vertices.

A property of msf's has been proven in a previous paper, as a basis of a linear time algorithm to build msf's in chain graphs. The same property is used here, to prove that msf's for binary trees can also be built in linear time, and an outline of the algorithm is given.

INTRODUCTION

Given a graph $G=(V,E)$, and a weight $w(e)\in Z^+$ for each $e\in E$, a spanning forest (or forest cover) is a subgraph $G'=(V,E')$, $E'\subseteq E$, consisting of (one or more) disjoint trees, each of which has at least two vertices. A minimal spanning forest (msf) is such that the sum of all $w(e)$, for $e\in E'$, is minimal [1,2]. This concept is an extension of the minimal spanning tree, however, msf is not necessarily a subgraph of such a tree.

The problem of determining an msf is solvable in $O(|V^3|)$ steps, as a special case of the degree constrained subgraph problem [3, Chpt. 6]. If the forest is constrained to include exactly k edges, the problem can be solved in $O(|V|^2\log|V|)$ steps by an extension of Kruskal's algorithm for minimal spanning trees [2]. Some related problems on partitioning G into forests are instead intractable (see, for example, problems GT14 and ND10 in [4]).

A graph and one of its msf's, are shown in the first two parts of Fig. 1, taken from [1]. It can be easily shown that any msf must be composed of stars. A basic result on msf's has been proven in [1], and is reported here in the following lemma. Let $F=(V,E_F)$ be an msf of G, and let $\overline{V}\subsetneq V$ be a subset of at least two vertices; $\overline{G}\subsetneq G$ be the subgraph induced on G by $\overline{V}$; $\overline{F}\subsetneq F$ be the subgraph induced on F by $\overline{V}$. Furthermore, let $\overline{\overline{V}}\subsetneq\overline{V}$ be the subset of all the vertices of $\overline{V}$ adjacent in F to vertices in $V-\overline{V}$. (I.e., for any $\overline{\overline{v}}\in\overline{\overline{V}}$ there is a $v\in V-\overline{V}$ with $(\overline{\overline{v}},v)\in E_F$. Obviously $\overline{\overline{V}}$ may be void.)

Lemma: If for any $\overline{\overline{v}}\in\overline{\overline{V}}$ there is a $\overline{v}\in\overline{V}$ such that $(\overline{\overline{v}},\overline{v})\in E_F$, then $\overline{F}$ is an msf of $\overline{G}$.

In Fig. 1, the set of vertices of G is restricted to $\overline{V}=\{B,C,D,N,H\}$. We have $\overline{\overline{V}}=\{N,H\}$, since (N,A), $(H,M)\in E_F$. Both N and H have connections in E_F with vertices in V, therefore, $\overline{F}$ is an msf of $\overline{G}$ by the lemma.

---

[*] This work has been supported in part by Consiglio Nazionale delle Ricerche of Italy.

[**] On leave from the University of Pisa, Italy.

In the present paper, we consider the msf problem for a restricted family of graphs, namely, binary trees. The lemma will be used to prove that msf's in binary trees can be built in $O(|V|)$ steps, by proving some additional properties and giving an outline of the algorithm.

PROPERTIES OF MSF'S OF BINARY TREES

Let our graph be a perfectly balanced binary tree $T=(V,E)$, that is:

$V=\{v_1,v_2,\dots,v_{2^h-1}\}$, $E=\{e_2,e_3,\dots,e_{2^h-1}\}$, $e_i=(v_{\lfloor i/2\rfloor}, v_i)$ for all i.

To present our results, we need some new notation. For $1\leq i\leq 2^h-1$, let $\Sigma_i\subseteq \{1,2,\dots,2^h-1\}$ be the subset of vertex indices such that:

$\Sigma_i=\{i,2i,2i+1,4i,4i+1,4i+2,4i+3,\dots,2^k i,2^k i+1,\dots,2^k i+2^k-1\}$, with $2^k i\geq 2^{h-1}$. Then, let $T_i=(V_i,E_i)$ be the subtree of T with root v_i, where $V_i=\{v_j:j\in\Sigma_i\}$, $E_i=\{e_j:j\in\Sigma_i-\{i\}\}$.

Denote by $F_i=(V_i,E_{F_i})$ an msf of T_i; and denote by $S_j(F_i)$, $j\in\Sigma_i$, the subgraph induced on F_i by the subset of vertices V_j. First we pose an assumption to simplify the discussion (this assumption will be removed in the last section).

Assumption 1: For any T_i there exists exactly one F_i, $1\leq i\leq\frac{n-1}{2}$. (In particular F is unique.)

We now state four corollaries of the lemma of section 1, applied to binary trees, whose meanings and proofs can be checked in Fig. 2. All these corollaries are valid under Assumption 1.

Corollary 1: If, for a given k, $e_{2i}\in E_{F_k}$, and/or $e_{2i+1}\in E_{F_k}$, then $S_i(F_k)=F_i$.

Proof: It follows from the lemma, applied to the restriction $S_i(F_k)$ induced of F_k by $\overline{V}=V_i$. (Here, the only possible vertex in $\overline{V}$ is $\overline{v}=v_i$; $\overline{v}=v_{2i}$ or $\overline{v}=v_{2i+1}$, and $(\overline{v},v)=e_{2i}$, or $(\overline{v},v)=e_{2i+1}$, exists in E_{F_k}.) Therefore $S_i(F_k)=F_i$ is an msf for T_i by the lemma, hence, is the only msf F_i by assumption 1. □

Corollary 2: If $e_i,e_{2i}\in E_{F_{\lfloor i/2\rfloor}}$, or $e_i,e_{2i+1}\in E_{F_{\lfloor i/2\rfloor}}$, then $S_i(F_k)=F_i$ for all k such that $i\in\Sigma_k$.

Proof: We refer to the case $e_i,e_{2i}\in F_{\lfloor i/2\rfloor}$. (The other case, e_i, $e_{2i+1}\in F_{\lfloor i/2\rfloor}$ can be treated symmetrically.) First we prove that $e_{2i}\in E_{F_k}$, for all k such that $i\in\Sigma_k$. In fact, if $e_{2i}\notin E_{F_k}$, then $e_i\in E_{F_k}$, or $e_{2i+1}\in E_{F_k}$, to provide an incident edge to vertex v_i. Both cases lead to a contradiction, namely: (a) $e_i\in E_{F_k}$, then $S_{\lfloor i/2\rfloor}(F_k)=F_{\lfloor i/2\rfloor}$ by corollary 1, against the hypothesis that $e_{2i}\in F_{\lfloor i/2\rfloor}$; (b) $e_{2i+1}\in E_{F_k}$, then $S_i(F_k)=F_i$ by corollary 1, thus implying that $e_{2i}\notin E_{F_i}$; however, e_{2i} must belong to E_{F_i}; in fact $e_{2i}\in E_{F_{\lfloor i/2\rfloor}}$ by hypothesis, then $S_i(F_{\lfloor i/2\rfloor})=F_i$ by corollary 1. In conclusion $e_{2i}\in E_{F_k}$, hence $S_i(F_k)=F_i$ by corollary 1. □

Corollary 2 states that if the two consecutive edges e_i,e_{2i} (or e_i, e_{2i+1}) are contained in $F_{\lfloor i/2\rfloor}$, then the subgraph $S_i(F_{\lfloor i/2\rfloor})$ is immutable in the msf's F_k for all the subtrees T_k containing T_i.

Corollary 3: If $e_i\notin E_{F_{\lfloor i/2\rfloor}}$, then $e_i\notin E_{F_k}$ and $S_i(F_k)=F_i$, for all k such that $i\in\Sigma_k$.

Proof: By contradiction we prove that $e_i \notin E_{F_k}$. In fact, if $e_i \in E_{F_k}$ then $S_{\lfloor i/2 \rfloor}(F_k)=F_{\lfloor i/2 \rfloor}$ by corollary 1, against the hypothesis of this corollary that $e_i \notin E_{F_{\lfloor i/2 \rfloor}}$. We can now state that $e_{2i} \in E_{F_k}$ or $e_{2i+1} \in E_{F_k}$, since an edge must be incident to vertex v_i in F_k. Therefore $S_i(F_k)=F_i$ by corollary 1. □

Corollary 3 states that if an edge e_i, incident to the root of $T_{\lfloor i/2 \rfloor}$, is not contained in the msf $F_{\lfloor i/2 \rfloor}$, then such an edge is immutably not contained in the msf's F_k of all the subtrees T_k containing T_i; and the subgraph $S_i(F_{\lfloor i/2 \rfloor})$ is immutably contained in all such F_k.

Corollary 4: If $e_{2i}, e_{2i+1} \in E_{F_i}$, then $S_i(F_k)=F_i$ for all k such that $i \in \Sigma_k$.

Proof: First note that $e_{4i}, e_{4i+1}, e_{2(2i+1)}, e_{2(2i+1)+1} \notin E_{F_i}$, since any of them, together with e_{2i}, e_{2i+1}, would generate a non-star subgraph of F_i. Then, by corollary 3, it is easily seen that $S_{2i}(F_k)=S_{2i}(F_i)$ and $S_{2i+1}(F_k)=S_{2i+1}(F_i)$. Since such subgraphs do not have incident edges to their vertices v_{2i} and v_{2i+1} then every F_k must include e_{2i} and e_{2i+1}. Hence $S_i(F_k)=F_i$. □

Corollary 4 states that if the two edges e_{2i}, e_{2i+1}, incident to the root v_i of T_i, are contained in F_i, then such an msf is an immutable portion of the msf's F_k of all the subtrees T_k containing T_i.

Consider, for example, the tree of Fig. 3, (note that such a tree fulfills assumption 1) and the msf's shown in the figure. The msf's of the subtrees with root at the last but one level ($F_{15},\ldots,F_8$ in the figure), must trivially coincide with such subtrees. The shape of the other msf's depends on the edge weights. By inspecting F_4, we note that e_8, e_{16}, $\in E_{F_4}$, then $S_8(F_k)=F_8$ for k=1,2,4,8 by corollary 2. That is, F_8 is an immutable subgraph of the msf's $F_{k=1,2,4,8}$ (this result is obvious in this case). A similar reasoning holds for $F_9,\ldots,F_{15}$.

By inspecting F_2 we note that $e_4 \notin E_{F_2}$, then $e_4 \notin E_{F_k}$ and $S_4(F_k)=F_4$ for k=1,2,4 by corollary 3. That is F_4 is an immutable subgraph of the msf's $F_{k=1,2,4}$. Finally, in F_3 we have $e_6, e_7 \in E_{F_3}$, then $S_3(F_k)=F_3$ for k=1,3 by corollary 4. That is F_3 is an immutable subgraph of $F_{k=1,3}$.

CONSTRUCTION OF AN MSF

The msf $F=F_1$ for T can be determined through the sequence of msf's $F_r, F_{r-1}, \ldots F_1$, where $r=2^{h-1}-1$ (recall that $n=2^h-1$ is the number of vertices in T). That is, through the sequence of msf's for the subtrees whose roots are internal nodes of T. Assumption 1 on the uniqueness of all F_i is still considered.

As already noticed in Fig. 3, the msf's $F_{2^{h-1}-1}, F_{2^{h-1}-2}, \ldots, F_{2^{h-2}}$ are known, in fact, they coincide with $T_{2^{h-1}-1}, T_{2^{h-1}-2}, \ldots T_{2^{h-2}}$ respectively. Assume now that the msf F_k for T_k, $k<2^{h-2}-1$, is the current msf to be determined. By hypothesis all the msf's $\overline{F}_j$ for the subtrees T_j of $T_k (j \in \Sigma_k - \{k\})$ have been already determined. However, some of these F_j will be recognized as immutable subgraphs of F_k. We now prove that, under assumption 1, the presence of immutable subgraphs restricts the possible configuration of F_k to five only.

We need some additional notation. For an index value j, let $\overline{j}=j+1$ if j is even, $\overline{j}=j-1$ if j is odd (hence, e_j and $e_{\overline{j}}$ are the edges connecting vertex $v_{\lfloor j/2 \rfloor}$ with its sons). In any subtree T_{h_1}, $h_1 \leq 2^{h-1}-1$, (see Fig. 4) there is a critical path $\pi_{h_1}=\{v_{h_1},v_{h_2},\ldots,v_{h_m}\}$, where $\lfloor h_i/2 \rfloor = h_{i-1}$ for $2\le i\le m$, and $m\ge 1$, such that:

1. v_{h_m} is the only vertex in π_{h_1} such that:

   a. $e_{h_m}, e_{2h_m} \in E_{F_{h_{m-1}}}$ or $e_{h_m}, e_{2h_m+1} \in E_{F_{h_{m-1}}}$,

or

   b. $e_{2h_m}, e_{2h_m+1} \in E_{F_{h_m}}$,

therefore F_{h_m} is an immutable subgraph of the msf's for all subtrees containing T_{h_m} by corollary 2 (case a) or corollary 4 (case b);

2. $e_{h_{i+1}} \in E_{F_{h_i}}$ and $e_{\bar{h}_{i+1}} \notin E_{F_{h_i}}$, for $1\le i\le m-1$,

therefore $F_{\bar{h}_{i+1}}$ is an immutable subgraph of the msf's for all subtrees containing T_{h_i}, and $e_{\bar{h}_{i+1}}$ does not belong to such msf's, by corollary 3.

A vertex v_{h_m} satisfying point 1 always exists, since $v_{2^{h-1}-1}$, $v_{2^{h-1}-2}$ ..., $v_{2^{h-2}}$ all satisfy point 1 (b). As a limit case, we may have e_{h_2}, $e_{\bar{h}_2} \in E_{F_{h_1}}$, hence $\pi_{h_1}=\{v_1\}$ (i.e., m=1), and the only point 1 (b) applies.

Let us now define the set of all edges in the immutable subgraphs found along π_{h_1}, as:

$$\Theta_{h_1} = \begin{cases} E_{F_{h_m}} \cup E_{F_{\bar{h}_m}} \cup E_{F_{\bar{h}_{m-1}}} \cup \ldots \cup E_{F_{\bar{h}_2}}, & \text{for } m>1, \\ E_{F_{h_1}}, & \text{for } m=1. \end{cases}$$

Furthermore, let:

$$\Gamma_{h_1} = \{e_{h_{m-2j}} : j\ge 0,\ h_{m-2j}\ge h_2\}, \qquad \gamma_{h_1} = \sum_{e\in\Gamma_{h_1}} w(e);$$

$$\Delta_{h_1} = \{e_{h_{m-2j-1}} : j\ge 0,\ h_{m-2j-1}\ge h_2\}, \qquad \delta_{h_1} = \sum_{e\in\Delta_{h_1}} w(e);$$

and let:

$$\Lambda_{h_1} = \Gamma_{h_1},\ \bar{\Lambda}_{h_1} = \Delta_{h_1},\ \text{if } e_{h_2} \in \Gamma_{h_1};$$

$$\Lambda_{h_1} = \Delta_{h_1},\ \bar{\Lambda}_{h_1} = \Gamma_{h_1},\ \text{if } e_{h_2} \in \Delta_{h_1}.$$

We can now prove that, under assumption 1, the configuration of any F_k depends on the critical paths π_{2k}, π_{2k+1} of its sons. In particular, E_{F_k} takes the configuration of minimum weight among the following (see Fig. 5):

$$E_{F_k} = \Theta_{2k} \cup \Theta_{2k+1} \cup \begin{cases} \{e_{2k}\} \cup \Gamma_{2k} \cup \Lambda_{2k+1} & \text{(a)} \\ \{e_{2k}\} \cup \Delta_{2k} \cup \Lambda_{2k+1} & \text{(b)} \\ \{e_{2k+1}\} \cup \Lambda_{2k} \cup \Gamma_{2k+1} & \text{(c)} \\ \{e_{2k+1}\} \cup \Lambda_{2k} \cup \Delta_{2k+1} & \text{(d)} \\ \{e_{2k}, e_{2k+1}\} \cup \bar{\Lambda}_{2k} \cup \bar{\Lambda}_{2k+1} & \text{(e)} \end{cases}$$

In other words, we state that, besides all the edges in Θ_{2k}, Θ_{2k+1} of the immutable subgraphs, F_k contains alternating edges on π_{2k}, π_{2k+1}.

To prove this statement, note that each vertex $v_{h_i} \in \pi_{2k}$ (or π_{2k+1}) must be covered in F_k by edge e_{h_i} or $e_{h_{i+1}}$ laying on π_{2k} (Fig. 4), since the other edge $e_{\bar{h}_{i+1}}$ incident to v_{h_i} does not belong to $E_{F_{h_i}}$ by definition of critical path, hence, $e_{\bar{h}_{i+1}} \notin E_{F_k}$ by corollary 3. Moreover, no two consecutive edges e_{h_i}, $e_{h_{i+1}}$ laying on π_{2k} (or π_{2k+1}) can belong to E_{F_k}. In fact, if $e_{h_i}, e_{h_{i+1}} \in E_{F_k}$, then $S_{h_{i-1}}(F_k)=F_{h_{i-1}}$ by corollary 1, hence $e_{h_i}, e_{h_{i+1}} \in E_{F_{h_{i-1}}}$. This contradicts the definition of critical path where only the final vertex v_{h_m} is such that $F_{h_{m-1}}$ has two consecutive edges leaving from the root (point 1(a) of the definition).

Once F_k is determined, as the configuration of minimum weight among (a),(b),(c),(d), and (e), the new critical path π_k is trivially formed. For example, we have in Fig. 5:

Case a: $\pi_k = \{v_k, v_{2k}\}$ (Corollaries 2 and 3)

Case b: $\pi_k = \{v_k\} \cup \pi_{2k}$ (Corollary 3)

Case e: $\pi_k = \{v_k\}$ (Corollary 4)

Two similar situations exist for cases c and d.

Let us now discuss how to build the msf F_1. For brevity, we will not formally give the algorithm, rather we indicate the lines along which such an algorithm can be built.

There is no need to build the sequence of all F_k, explicitly, for $k=2^{h-1}-1, \ldots, 2, 1$, but only to build the immutable subgraphs and critical path for each F_k.

At step k, the sets Θ_{2k}, Θ_{2k+1}, the paths π_{2k}, π_{2k+1}, and the values of γ_{2k}, γ_{2k+1}, δ_{2k}, δ_{2k+1}, are all known. Through these values the weights of the five possible configurations (a) to (e) of E_{F_k} are computed and compared, to select the configuration of minimum weight. (Note that this configuration is not explicitly built.) Then, Θ_k, π_k, γ_k and δ_k are built or computed by updating the configurations, or values, of the same entities for 2k, 2k+1. When v_1 is reached, F_1 is explicitly built by determining E_{F_1} according to (a),(b),(c),(d), or (e).

In conclusion, the construction of F_1 requires $2^{h-2}-1$ steps, that is $O(|V|)$ steps. At the generic step k, the following operations are done:

1. Determination of the minimum weight configuration. This clearly requires constant time.

2. Determination of Θ_k. This is obtained by combining a number of (immutable) Θ_i previously built, with new edges. However, every edge of T is taken at most once in the overall process, since, when taken, it becomes a part of an immutable portion of F_1. Therefore, the overall time needed to build all Θ_k is of $O(|E|)=O(|V|)$.

3. Determination of π_k (see the examples given in connection with Fig. 5). We have one of the following cases:

a. $\pi_k=\{v_k,v_{2k}\}$, or $\pi_k=\{v_k,v_{2k+1}\}$, or $\pi_k=\{v_k\}$, each determined in constant time

b. $\pi_k=\{v_k\} \cup \pi_{2k}$, or $\pi_k=\{v_k\} \cup \pi_{2k+1}$, that are also determined in constant time, since π_{2k} and π_{2k+1} have been determined in previous steps.

4. Evaluation of γ_k and δ_k. These are weights of edges in π_k, hence they depend on the configuration of π_k. In the cases listed in point 3a. above, γ_k and δ_k are immediately computed in constant time. In the cases listed in point 3b. γ_k and δ_k are also determined in constant time, from the values of γ_{2k}, γ_{2k+1}, δ_{2k}, δ_{2k+1} previously evaluated.

Therefore, the overall time required by the algorithm is of $O(|V|)$, that is linear in the number of vertices. One can easily note that the algorithm is also linear in space.

GENERALIZATIONS

Assumption 1 was posed to simplify our presentation. We now prove that, without this assumption, the proposed algorithm still holds, with a minor modification.

As in the original position, let the tree T have weights $w(e_i)\in Z^+$ for all i. As known, without assumption 1 the msf for T is generally not unique. Consider now the new tree T', where T'=T except for the edge weights, that are changed as follows: $w'(e_i)=w(e_i)+1/2^i$, for $2\leq i\leq n$.

It can be immediately verified that T' fulfills assumption 1 (no two distinct subsets of its edges have the same total weight); and that, for any given k, the msf F'_k for the subtree T'_k of T' is also an msf for the subtree T_k of T.

Therefore, after having removed assumption 1, the algorithm can be modified as follows, to be applied to T. At the generic step k, several minimum weight configurations may exist among (a),(b),(c),(d), and (e). Choose the one that appears first in the following order: (c) or (d); (a) or (b); or (e); where, if (c) and (d) have the same minimum weight, then select the one that does not include any edge leaving from v_{2k+1}, besides e_{2k+1}. Similarly, if (a) and (b) have the same minimum weight, select the one that does not include any edge leaving from v_{2k}, besides e_{2k}. In fact, the above discipline corresponds to applying the algorithm of the previous section to T'. At each step this algorithm would select the same configuration, since such a configuration has the smallest total weight if computed with w'. The modified version of the algorithm has obviously the same linear time and space complexity of the original one.

It can be proven that similar arguments hold for all the perfectly balanced degree constrained (other than binary) trees, where an extension of the above algorithm determines msf's in linear time. Finally, the algorithm can be further extended to all unbalanced degree constrained trees. For the limit case of chains, a linear algorithm has been given in [1].

REFERENCES

[1] F. Luccio and L. Pagli. "A linear algorithm to determine minimal spanning forests in chain graphs," *Information Processing Letters*, (1982, to appear).

[2] L.J. White. "Minimum covers of fixed cardinality in weighted graphs," *Siam J. Appl. Math.*, 21, 1971, pp. 104-113.

[3] E.L. Lawler. *Combinatorial Optimization: Networks and Matroids*, New York: Holt, Rinehart and Winston, 1976.

[4] M.R. Garey and D.S. Johnson. *Computers and Intractability*, San Francisco: W.H. Freeman and Co., 1979.

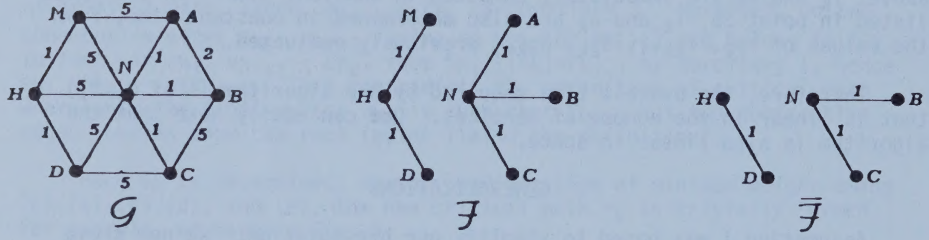

M
5
A
1
5
N
1
2
H
5
1
B
1
5
1
5
D
5
C
G
M
A
1
1
H
N
1
B
1
1
D
C
F
H
N
1
B
1
1
D
C
$\overline{F}$

Fig. 1. (from [1]). A graph G; an msf $\underline{F}$ of G; the subgraph $\overline{F}$ induced on F by $\overline{V}$={B,C,D,N,H}

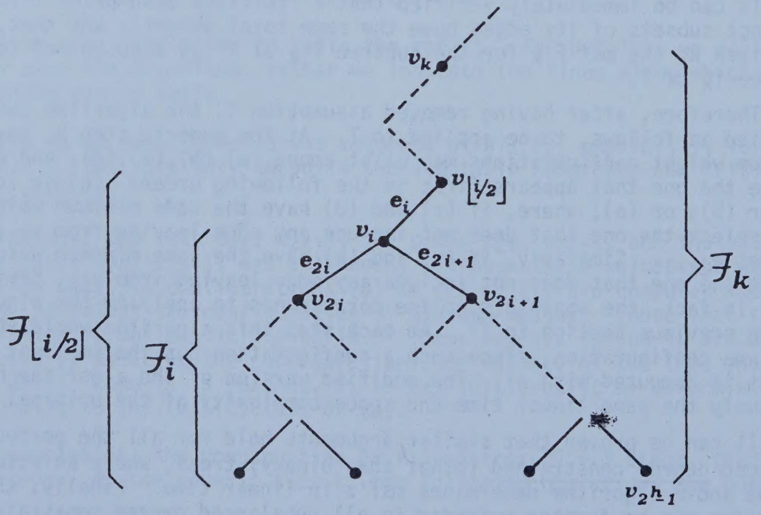

v_k
$v_{\lfloor i/2 \rfloor}$
e_i
v_i
e_{2i}
e_{2i+1}
v_{2i}
v_{2i+1}
$F_{\lfloor i/2 \rfloor}$
F_i
F_k
v_{2^h-1}

Fig. 2. Illustration of corollaries 1,2,3, and 4

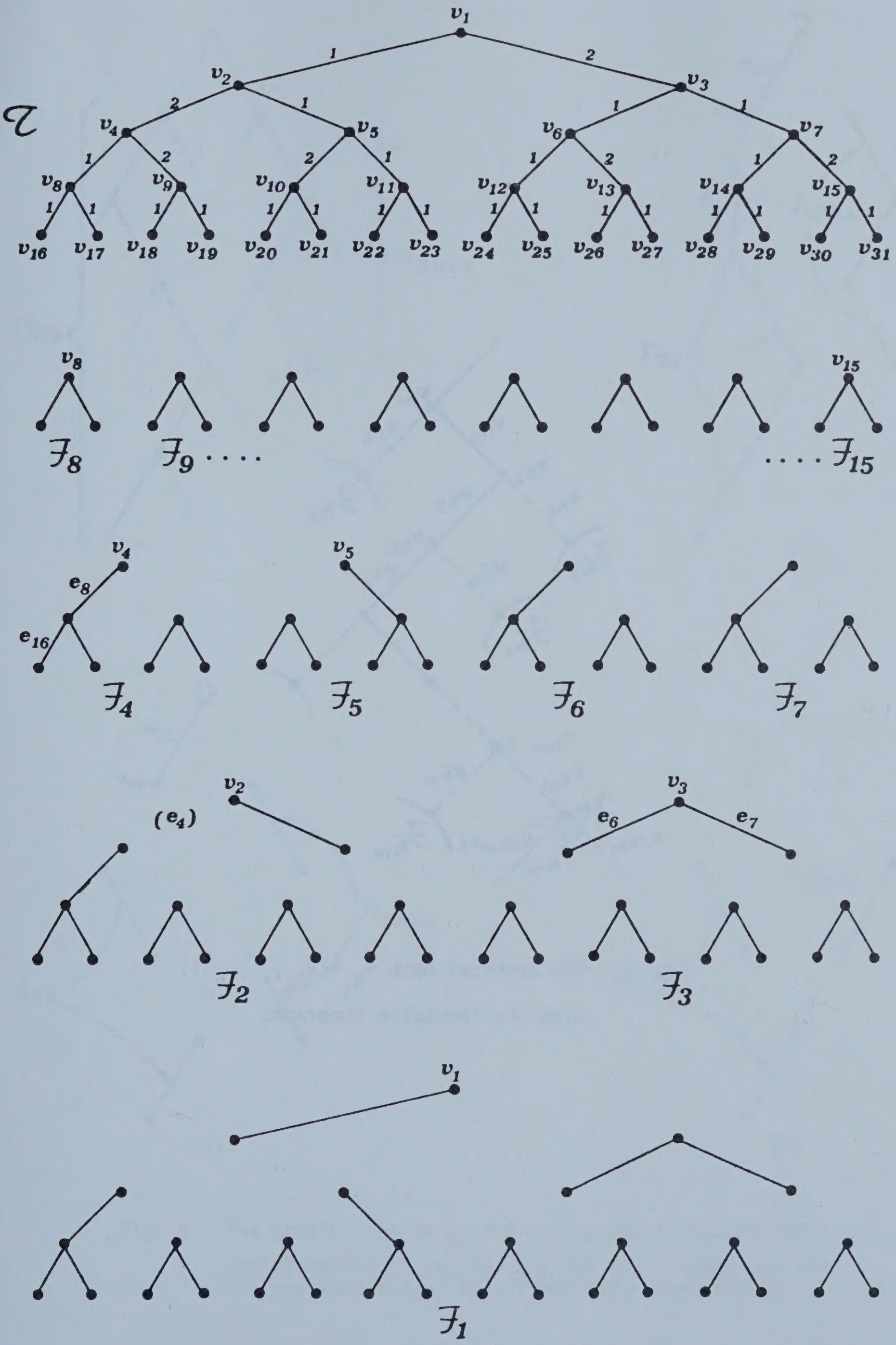

v_1
1
2
v_2
v_3
$\mathcal{T}$
2
1
1
1
v_4
v_5
v_6
v_7
1 2 2 1 1 2 1 2
v_8 v_9 v_{10} v_{11} v_{12} v_{13} v_{14} v_{15}
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
v_{16} v_{17} v_{18} v_{19} v_{20} v_{21} v_{22} v_{23} v_{24} v_{25} v_{26} v_{27} v_{28} v_{29} v_{30} v_{31}
v_8
v_{15}
$\mathcal{F}_8$
$\mathcal{F}_9$ ....
.... $\mathcal{F}_{15}$
v_4
e_8
e_{16}
v_5
$\mathcal{F}_4$
$\mathcal{F}_5$
$\mathcal{F}_6$
$\mathcal{F}_7$
v_2
(e_4)
v_3
e_6
e_7
$\mathcal{F}_2$
$\mathcal{F}_3$
v_1
$\mathcal{F}_1$

Fig. 3. A binary tree T and the msf's for some subtrees $T_{j\geq 1}$

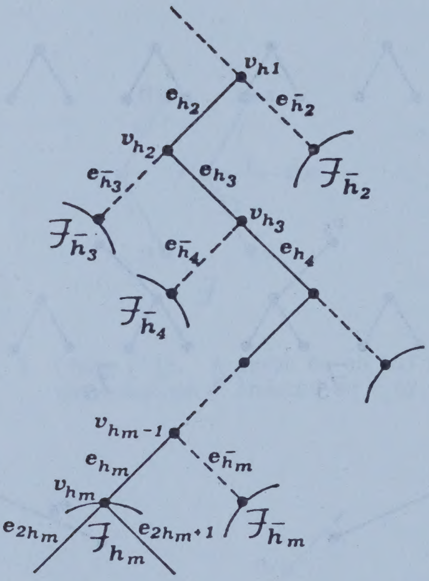

v_{h1}
e_{h_2}
$e_{\bar{h}_2}$
v_{h_2}
$\mathcal{F}_{\bar{h}_2}$
$e_{\bar{h}_3}$
e_{h_3}
$\mathcal{F}_{\bar{h}_3}$
v_{h_3}
$e_{\bar{h}_4}$
e_{h_4}
$\mathcal{F}_{\bar{h}_4}$
$v_{h_{m-1}}$
e_{h_m}
$e_{\bar{h}_m}$
v_{h_m}
e_{2h_m}
$\mathcal{F}_{h_m}$
e_{2h_m+1}
$\mathcal{F}_{\bar{h}_m}$

Fig. 4. The critical path $\pi_{h_1}=\{v_{h_1},..,v_{h_m}\}$, with the immutable subgraphs

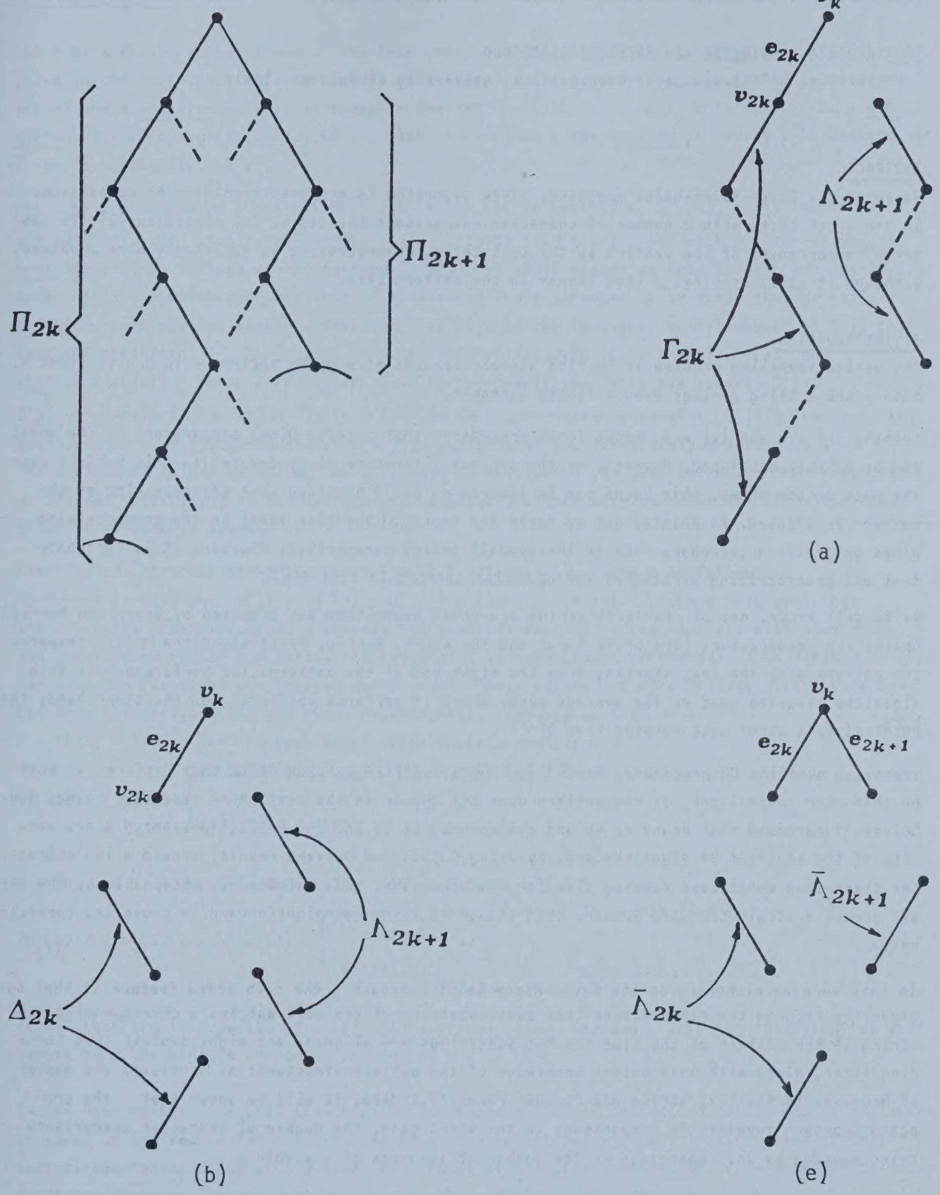

Π_{2k}
Π_{2k+1}
v_k
e_{2k}
v_{2k}
Λ_{2k+1}
Γ_{2k}
(a)
v_k
e_{2k}
v_{2k}
Λ_{2k+1}
Δ_{2k}
(b)
v_k
e_{2k}
e_{2k+1}
$\bar{\Lambda}_{2k+1}$
$\bar{\Lambda}_{2k}$
(e)

Fig. 5. The critical paths π_{2k} and π_{2k+1}, and three possible configurations: (a),(b),(e) for E_{F_k}. Cases (c) and (d) are symmetrical to (a) and (b), respectively.

ELABORATING ON THE BOYER-MOORE-GALIL STRING SEARCHING STRATEGIES[*]

ALBERTO APOSTOLICO and RAFFAELE GIANCARLO
Istituto di Scienze dell' Informazione, University of Salerno, Italy

ABSTRACT

Based on the Boyer-Moore-Galil approach, a new algorithm is proposed involving 6n comparisons in the worst case, with a number of character comparisons bounded by 2n, regardless of the number of occurrences of the pattern in the textstring. Preprocessing is admittedly more involved, although it still requires a time linear in the pattern size.

1. Introduction

The string searching problem is to find all occurrences of a given pattern y in a given text x, both y and x being strings over a finite alphabet.

Letting $|x| = n$ and $|y| = m$, brute force procedures that involve $O(nm)$ comparisons in the worst can be quickly developed. However, as the copious literature [1-6] devoted to this subject over the past decade shows, this bound can be lowered to $O(n)$, provided some preprocessing of the pattern is allowed. As pointed out by Boyer and Moore [3] the time spent in the preprocessing plays generally a secondary role in the overall design perspective. However, it is fortunate that all preprocessing strategies set up so far perform in time $O(m)$.

As is well known, one of the first string searching algorithms was proposed by Boyer and Moore [3] Unlike its predecessors (the brute force and the Knuth, Morris, Pratt algorithm [1]) it compares the pattern with the text starting from the right end of the pattern. The performance of this algorithm is quite good on the average case, where it performs in $O(n/m)$. On the other hand, the BM displays a worst case running time $O(n^2)$.

Improving over the BM procedure, Knuth [1] set up a modified version of it that performs at most 6n character comparisons, if the pattern does not appear in the text. More recently, Guibas and Odlyzko [7] narrowed that bound to 4n and conjectured it is 2n. Zvi Galil [6] presented a new version of the modified BM algorithm and, by using Guibas and Odlyzko result, showed a 10n character comparison worst case running time for his algorithm. This solution is obtainable by the former one in a straightforward manner, even though it is not straightforward to prove its correcteness.

In this work we elaborate on the Boyer-Moore-Galil approach ; the main added feature is that our algorithm records the circumstance that some substring of the text matches a corresponding substring of the pattern at the time the two substrings are aligned, and might exploit this recording later, along with an a priori knowledge of the pattern structure: as a result, the number of accesses to the text string are further reduced. Indeed, it will be shown that the proposed approach involves 6n comparisons in the worst case, the number of character comparisons being bounded by 2n, regardless of the number of istances of y within x .

On the other hand, preprocessing is admittedly more involved, as it rests on an auxiliary index (namely a suffix tree [8]) built out of the pattern : nevertheless, the time required by preprocessing is still linear in the lenght of y .

---

[*] This work was supported in part by the Italian Ministry of Education (National Council for the Mathematical Sciences) and by N.A.T.O. under Research Grant N° 039.82

2. Three shift functions

Let I be a finite alphabet and I^+ the free semigroup generated by I. A string y in I^+ of lenght $|y|=m$ is the concatenation of m symbols of I. Here it is assumed that y is stored in an array y(1,m) and a position in y is an index in the set P = (1,2, ... , m) ; w is said to be a substring of y if w = y(i)y(i+1)...y(j) , with $1 \leq i \leq j \leq m$. The reverse of string y is denoted by $\tilde{y}$ or, equivalently, by $y^{\sim}$.

Let j be a position in y and let B(j) denote the set of positions $j_1 , j_2, \dots, j_k$ such that $y(j_t)y(j_t+1)\dots y(j_t+m-j) = y(j)y(j+1)\dots y(m)$, $1 \leq t \leq k$ and $j_t \neq j$. For each j , the neighbour position of j is defined as the maximum in B(j). We shall assume at this stage , and justify later, that the neighbour positions of indexes in P are arranged in an array that we call NEIGH(j), with obvious meaning. Moreover, let C(j) be the (possibly empty) subset of B(j) that contains positions $j'_1 , j'_2 , \dots , j'_{k'}$ such that $y(j'_t)\dots y(j'_t+m-j) = y(j)\dots y(m)$ and $y(j'_t-1) \neq y(j-1)$, $1 \leq t \leq k$. If $C(j) \neq \emptyset$, we further partition it in the subsets $C(j)/a_1 , \dots , C(j)/a_i$, where $i \leq |I|$ and a_t is in I for $1 \lessgtr t \lessgtr i$. For each value of t, $C(j)/a_t$ collects the positions in y of the instances of w = y(j)...y(m) that are immediately preceded by a_t . The maximum of $C(j)/a_t$ is called the a_t-leader at position j .We also introduce the array LEAD as follows : LEAD(a,j) is the a-leader at position j . Once more we defer the efficient construction of LEAD to section 4.

Finally, we introduce three functions (shift functions) G , F and H as follows :
G: $P \rightarrow N*P$ is such that $G(j) = (g_1(j) , g_2(j)) = (2m-s-j , m-s+1)$, where s is such that $y(1)\dots y(s) = y(m-s+1)\dots y(m)$, whereas the quantity m-s+1-j is the smallest with varying s. The experienced reader has surely noticed that G is reminiscent of the dd' table defined in |1|. The next two functions are instead quite novel. Their values are more or less directly derivable from the arrays NEIGH and LEAD, respectively. Formally we have that :
$F = (f_1 , f_2) : P \rightarrow N*P$ is such that , with obvious meaning :

$$(f_1(j) , f_2(j)) = \begin{cases} (m-NEIGH(j)+1 , j) & \text{if } NEIGH(j) \neq \emptyset \\ (g_1(j)+1 , g_2(j)) & \text{otherwise.} \end{cases}$$

$H = (h_1 , h_2 , h_3) : I*P \rightarrow N*P*\{1,2\}$ is such that:

$$(h_1(a;j) , h_2(a,j) , h_3(a,j)) = \begin{cases} (m-LEAD(a,j)+1, j , 1) & \text{if } LEAD(a,j) \neq \emptyset \\ (G(j-1),2) & \text{otherwise.} \end{cases}$$

Recall that|1|q is a period of y iff $y(i) = y(i+q)$, for $1 \lessgtr i \lessgtr m-q$. In the following, we will denote by p the minimum among the periods of y .

In the next section, we will show how the functions and tools introduced so far can be exploited to speed up the search for all istances of y in a textstring x . Full justification of this last circumstance, along with a detailed account of the preprocessing phase is deferred to later sections. Occasionally, we will also assume that testing whether or not a generic position t of y belongs to any of the sets B(j) or C(j)/a involves exactly one comparison. Also this assumption will be fully justified later .

3. The algorithm

Let y and x denote the pattern and the text, respectively, and assume that the functions F, H and G have been computed and stored so that their values are available when the arguments are specified. We start by describing the role played by the function H in the string matching process. Assume that, at some stage in the process, the pattern be aligned with the text in such a way that $y(i+1)\ldots y(m) = x(k+1)\ldots x(k+m-i-1)$ and $y(i) \neq x(k)$. Since there is a mismatch, the pattern must slide over the text rightward: this shift takes place under control of H as follows. Consider $H(x(k),i+1)$ and suppose first that $LEAD(x(k),i+1) \neq \emptyset$: this means that there is a position j in P such that $y(j-1) = x(k)$ and $y(j)\ldots y(j+m-i) = x(k+1)\ldots x(k+m-i-1)$. In this case, $h_1(x(k),i+1)$ yields the shift that will take the two strings to be aligned. As we show later, $h_2(x(k),i+1)$ and $h_3(x(k),i+1)$ are used to take memory that $x(k)\ldots x(k+m-i-1)=y(j-1)\ldots y(j+m-i)$.

On the other hand, if $LEAD(x(k),i+1) = \emptyset$ then there is no position in P such that the equality $y(j-1)\ldots y(j+m-i)=x(k)\ldots x(k+m-i-1)$ holds . In this case the pattern must move rightward over the text to the closest position where one of its prefixes fully matches a suffix of $x(k+1)\ldots x(k+m-i-1)$. This shift is controlled by H too. Indeed, $h_1(x(k),i+1)=g_1(i)$ yields the shortest shift by which $y(1)\ldots y(s)=x(t)\ldots x(t+s-1)$, where $k<t$ and the quantity $m-s+1-j$ is the smallest with varying s (again, h_2 and h_3 are used to record this match) .

To take record of the circumstance that some substring of the pattern matches with a substring of the text, we introduce the auxiliary array x'(1,n) . (Actually, there exist straightforward O(m) space implementations for such an auxiliary table. However, this would unnecessarily complicate the exposition of the method). Each entry in x' is so formatted as to contain three fields that shall be referred to as label , position and set , respectively. The values that are found stored in these fields depend on which case of the H function was used following the detection of a mismatch.

To explain this point, let head be the position of the text aligned with the last position of the pattern in the process that was being discussed above. Then, if $x(k) \neq y(i)$ and $LEAD(x(k),i+1)$ is not zero,the values $h_2(x(k),i+1)$ and $h_3(x(k),i+1)$ will be recorded in the position and label field of x'(head) while the set field points to $C(i)/x(k)$. That happens if $LEAD(x(k),i+1)=\emptyset$ too, but the set field is meaningless in this case. In both circumstances the variable head assumes the value $k+h_1(x(k),i+1)$ and the process will start again by comparing x(head) with y(m).

Once these quantities have been stored in x', they can be used to save on character comparisons in the following way. With reference to Fig. 1 , let the pattern be aligned with the text and let x(r) and y(j) be the characters to be compared next. Before actually comparing them, it is checked as to whether x'(r) is not empty. If x'(r) is empty, then the comparison must take place and the pattern matching process goes on accordingly. On the other hand, assume that x'(r) is not empty.Then one of the following cases applies:

Case 1 The value 1 is found stored in the label field of x'(r), while the values $C(i)/x(k)$ $(k = r-m+i-1)$ and i are found stored in the set and position fields of x'(r) , respectively .

Let j be the position of the pattern currently aligned with position r of the text. Two subcases must be considered:

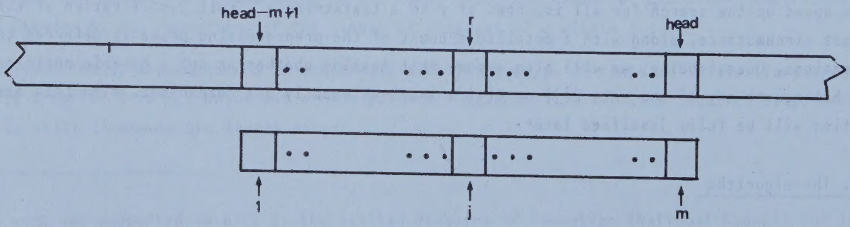


Fig. 1

1a) The difference $j-(m-i+2) \geq 0$, i.e. the substring of the text x(r-(m-i+1),r) is completely aligned with a substring of y (Fig. 2.a) .

1b) The difference $j-(m-i+2) < 0$, i.e. a suffix of the substring x(r-(m-i+1),r) is aligned with a prefix of y (Fig. 2.b) .

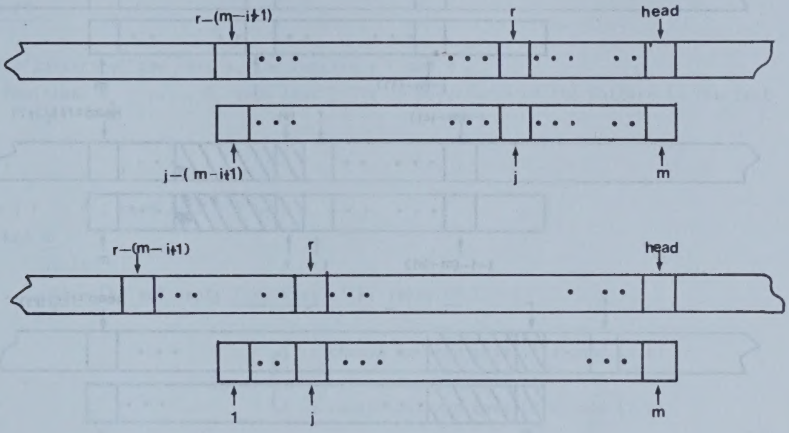


Fig. 2

Subcases (1a) and (1b)

If (1a) applies, then the following actions are in order:

1.a.1) If j-m+i belongs to C(i)/x(k) then the substring x(r-(m-i+1),r) is equal to y(j-(m-i+1),j) so the corresponding positions can be skipped (Fig. 3) and the pattern matching goes on by considering positions (j-(m-i+2)) and (r-(m-i+2) .

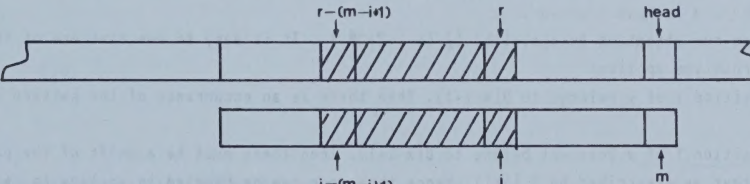


Fig. 3

¬x'(r) is not empty, x(r-(m-i+1),r) is of the form x(r-(m-i+1))y(i)...y(m).Since j-m+i is in C(i)/x(r-(m-i+1)), the shaded region may be skipped.

1.a.2) On the other hand, if j-m+i does not belong to C(i)/x(k) then the substrings x(r-(m-i+1),r) and y(j-(m-i+1),j) are different (Fig. 4a) . Thus there must be a shift of the pattern over the text. The value of such shift is found stored in $f_1(j+1)$. At this time $f_2(j+1)$ is copied in the position field of x'(head) and the label field of x'(head) will be set to 2 . The value $r+f_1(j+1)$ is assigned to head and the pattern matching process must be resumed by comparing x(head) with y(m). Indeed, at next step the position r+1 of x might be aligned with position t in P, where t is the neighbour position of j+1 (Fig. 4b) . If j+1 does not have a neighbour position , then the shift is such that a position r' of x, r+1≥r', results aligned with position 1 in y (Fig. 4c) . Note how the use of the F table avoids comparisons among characters that have already been examined.

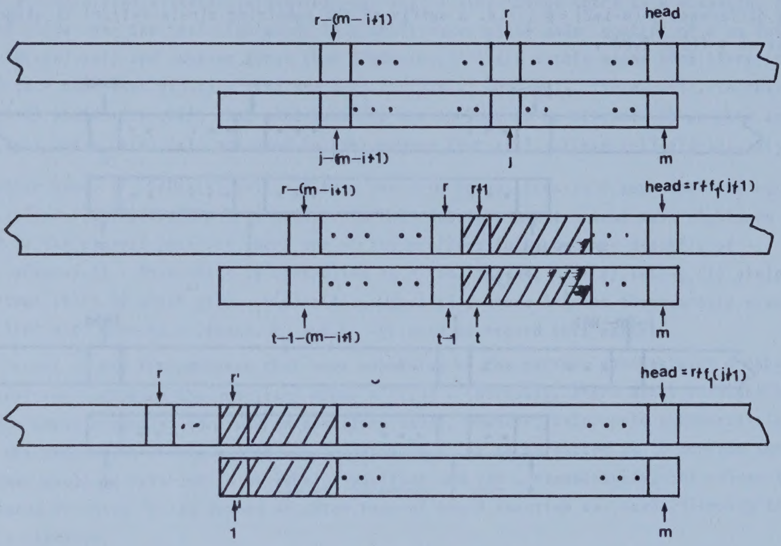


Fig. 4

Since j-m+1 is not in C(i)/x(r-(m-i+1)), then y(j-(m-i+1),j) is not equal to x(r-(m-i+1),j) . If t is the neighbour position of j+1, the shift made by means of f_1(j+1) leads from the situation in Fig. 4.a to that of Fig. 4.b . The shaded region in Fig. 4.b is equal to y(j+1)...y(m) . If j+1 has no neighbour position, then f_1(j+1) leads from the situation in Fig. 4.a to that of Fig. 4.c .The shaded region in Fig. 4.c is equal to y(j')...y(m) , j+1<j' .

We now turn our attention to case (1b) (j-(m-i+2)<0) . It is easy to see that one of the following alternatives applies:

1.b.1) Position 1 of y belongs to B(m-i+1). Then there is an occurrence of the pattern in the text.

1.b.2) Position 1 of y does not belong to B(m-i+1). Then there must be a shift of the pattern over the text as prescribed by f_1(j+1) (hence this case can be handled in analogy to case (1.a.2) discussed above).

Case 2 We have the following hypothesis: the label field of x'(r) is set to 2 and the position field contains integer i. Let j be the position of y aligned with position r of x. Again, this case splits in two subcases:

2.a) The difference j-(m-i+1)≥0 .

2.b) The difference j-(m-i+1)<0 .

Subcase (2.b) is identical to subcase (1.b) . If subcase (2.a) applies , then one of the following actions are undertaken:

2.a.1) If j-(m-i) belongs to B(i) then there is an occurrence of the pattern in the text.

2.a.2) If j-(m-i) does not belong to B(i) then an action analogous to case (1.a.2) is to be started.

This completes the discussion of case 2 . There is one more point to clarify, that is, what happens when an occurrence of the pattern in the text is detected. But clearly, if this is the case the label field of x'(head) is to be set to 2 and the position field will assume the value p+1 , while the variable head must be set to head = r+m+p . Finally, recall from |1| that : d(a) equals to $\min\left\{t / \ t=m \ \underline{\text{or}} \ (0 \leq t<m \ \underline{\text{and}} \ y(m-t) = a)\right\}$. We now synthetize the above discussion in form of a listing as follows :

Algorithm PM

Input : The pattern y, the text x, the functions F and H .
Output : Positions i_1 , ... , i_k such that i_j is an occurrence of the pattern in the text .
Method :

```
  Begin
   k := m ; r := 0 ; head := m ; j := m ;
   While k≤n do
            Begin
1)..........If x'(k) not empty Then Case x'(k) label Of :
                  1 : choose between case (1.a) and (1.b)
2)..............
                  2 : choose between case (2.a) and (2.b)

            Else Case ( y(j) = x(k)) Of :
                  False : If j = m Then k := k+d(x(k)) ; head := k ;
                                   Else k := h_1(x(k),j+1) ; x'(head) is set
                                        h_3(x(k),j+1),C(j)/x(k),h_2(x(k),j+1) ;
                                        head := k ;
3)..............
                  True  : j := j-1 ; k := k-1 ;
                          If j≤r Then "match at k+1" ; j := m ; r := m-p ;
                                  k := k+m+p ; head := k ; x'(head) is
                                  set to 2 , 0 , p ;
            End
  End
```

The same discussion legitimates the following :

Claim 1 The algorithm PM detects all and only the occurrences of y in x .

Analysis of algorithm PM

We now measure the performance of PM in terms of comparisons among quantities. Thus we will assume as meaningful the comparisons made at point 1) of the program listed earlier, togheter with the ones involved in the two Case structures. Because of the need ,during the following discussion, of a concise reference to these structures , we assigned a reference number to them in the above listing. It should be noted that PM does not allow more than n-m+1 shifts of the pattern over the text. Let us divide such shifts in two classes: the ones due to functions F and H and the ones due to d. Let k_1 be the number of the shifts belonging to this latter class. Each of these shifts can cause at most one character comparison ,since use of d function is due to a mismatch between y(m) and the corresponding character of the text. Moreover, each time the algorithm uses functions F or H then a position of x' is marked and this will avoid further exam of characters that have already been considered. Thus , the part of code referred as 3 is executed at most $n+k_1$ times. It is easy to see that step 1 is executed each time step 2 is executed plus each time step 3 is executed.We claim that step 2 cannot be taken more than $n-m+1-k_1$ times. Indeed, we can have at most $n-m+1-k_1$ positions of x' which are not empty; M_1 such positions will be analized at most once because there is agreement between the marked text substring and the corresponding pattern substring. On the other hand, M_2 positions of x' (not empty) can be analized more than once and they cause a shift of the pattern over the text until the marked text substring and the corresponding pattern substring do not coincide or such marked text substring is left behind the pattern. These latter M_2 positions can be charged with at most $n-m+1-k_1-M_1$ shifts, thus step 2 is entered at most $n-m+1-k_1$ times and step 1 is executed n+n-m+1 times at most. Let C denote the number of comparisons made each time step 2 is entered, thus it must be charged with $C(n-m+1-k_1)$ comparisons. By the analysis that preceded the algorithm, it results that C is at most 3, whence the total number of comparisons cannot excede the value $3(n-m+1-k_1)+2n-m+1+n+k_1 = 6n-4m-2k_1+4 < 6n$. The analysis made so far justifies the following :

Claim 2 The algorithm PM finds all occurrences of y within x in 6n comparisons at most, the number of character comparisons being bounded by 2n, regardless of the number of istances of y in x.

It should be remarked that, if at step t the algorithm shifts the pattern one position over the text, by means of H function, then at step t+1 the pattern must shift by at least 2 positions over the text. Thus the bound of Claim 2 is not tight.

4. Preprocessing

In this section we adress the problem of computing the functions F, G and H introduced in section 2. and justify the assumption that led to the algorithm of the last section. Specifically we left the following issues to be clarified:

1) The computation of the functions F and H as related to the arrays NEIGH and LEAD .
2) The computation of G .
3) The computation of the minimum period p of y .
4) The legitimation of the assertion that it is possible to check in one comparison whether a position t of y belongs to C(j)/a or B(j) .

For these porpuses we will make use of the suffix tree $T_{\tilde{y}}$ |8| associated with y . As is well known, the construction of $T_{\tilde{y}}$ requires O(m) steps . We refer to ref. |8| for the details of such construction as well as for the basic definitions, to which we shall entirely conform hereafter. In $T_{\tilde{y}}$, let u be the first node on the path from ROOT($T_{\tilde{y}}$) to leaf labeled 1 . We leave it to the reader as a simple exercise to show that the period p and the function G can be quickly computed by visiting T^u , the subtree of $T_{\tilde{y}}$ rooted at u . This takes care of points (2) and (3) above .

For each position j of y, it is clearly, with obvious meaning : $(y(j)...y(m))^{\sim} = \tilde{y}(1)...\tilde{y}(m-j+1)$ We will refer to i = m-j+1 (j) as to the conjugate position of j (i) , denoted S(j) (S(i)) . Given a set C(j)/a, let v be the extended locus of $(ay(j)...y(m))^{\sim}$ in $T_{\tilde{y}}$. From the definition of suffix tree it follows straightforwardly that:

Fact 1 A position t in y is in C(j)/a iff the leaf labeled S(t+(m-j)) is a descendant of v .

Fact 2 The a-leader at position j of y is S(t+(m-j)), where t is the minimum among the labels of the leaves of T^v , the subtree of $T_{\tilde{y}}$ rooted at v .

In the following we will refer to v shortly as to the locus of C(j)/a . We remark that leaf 1 cannot be a descendant of v . Notice also that v is the unique node in $T_{\tilde{y}}$ such that all and only the positions that are conjugate of t+(m-j) , t in C(j)/a , are found in T^v . It will be called the eigen node associated with such conjugate positions .

Let u_1 , u_2 , ... , u_k be the sequence of nodes on the path from ROOT($T_{\tilde{y}}$) to leaf labeled 1 . For each u_i , consider the subtrees of T^{u_i} , exept $T^{u_{i+1}}$, and let s_1 , ... , s_r be the labels of the leaves of such subtrees . We assign to each s_k , $1 \leq k \leq r$, the value $|u_i|$ of the number of symbols encountered on path from ROOT($T_{\tilde{y}}$) to node u_i , and refer to it as to the lenght of s_i . Given a set $B(j) \neq \emptyset$, let u_i be the extended locus of $(y(j)...y(m))^{\sim}$. By the definition of suffix tree it follows that:

Fact 3 A position t of y is in B(j) iff , denoting by d the lenght of S(t+m-j) , it is $d \geq m-j+1$

Moreover , letting $s(u_i)$ be min $\{s_1 , ... , s_r\}$, then we have:

Fact 4 Given a node u_i and a set $B(m-|u_i|+1)$ and denoted by t the minimum label among the leaves of $T^{u_{i+1}}$, the neighbour position of $m-|u_i|+1$ is $S(\min(s(u_i)+|u_i|-1 , t+|u_i|-1)$.

We are now in the position to account for point (1) above . The method for computing NEIGH and LEAD can be sketched as follows .

Visit the nodes u_k , ... , u_1 in ascending order and let u_i the node currently considered and v_1 , ... , v_s be its son nodes ,other than u_{i+1} . Visit T^{v_j} and set a pointer from each leaf to v_j . Let t_j be the minimum among the labels of such leaves and let a be the first symbol on the arc from u_i to v_j . Using Fact 2 above , set LEAD$(a,m-|u_i|+1)$ to the value $S(t+|u_i|-1)$. By Fact 4 , set also NEIGH$(m-|u_i|+1)$ equal to min $\{t_1 , ... t_s , \text{NEIGH}(m-|u_{i+1}|+1)\}$.

Whence LEAD and NEIGH are computed by a simple visit of T_y , in linear time . Notice that each of nodes v_1 , ... , v_s above is the locus of some nonempty set C(j)/a . Thus a table LOCUS(a,j) to access randomly such nodes can also be arranged during the visit . This construction is straightforward and its details are left to the reader .

We now turn to point (4) . In order to check whether t is in C(j)/a , simply access the eigen node v of S(t+m-j) via the pointer from the associated leaf . By Fact 1 , if v equals LOCUS(a,j), then t is in C(j)/a . Checking whether t is in B(j) is done by exploiting Fact 3 , which takes almost straightforward manipulation .

Figure 5. , on next page , provides an illustration of these access tecniques in connection with the pattern y = AABAAABABAABA . The only non empty B-type sets for the pattern are :

$B(10) = \{1 , 5\}$
$B(11) = \{2 , 6 , 8\}$
$B(12) = \{3 , 7 , 9\}$
$B(13) = \{1 , 2 , 4 , 5 , 6 , 8 , 10 , 11\}$

The C-type sets are :

$C(10)/A = \{5\}$
$C(11)/B = \{8\}$
$C(13)/A = \{11 , 6 , 5 , 2\}$

Node v is the eigen node for C(13)/A , so that pointers are directed from the leaves in the subtree T^v of $T_{\tilde{y}}$ to v (eigen nodes for the others are leaves 4 and 6 , respectively) . The a-leader of position 13 is the conjugate 11 of the minimum label 3 among the leaves of T^v . In order to check , say , whether 6 is in C(13)/A , it suffices to consider the conjugate 8 of 6 , access (randomly) leaf 8 whereby v via the pointer set up from this leaf . Since v is also pointed to by LOCUS(A,13) then 6 belongs to C(13)/A .

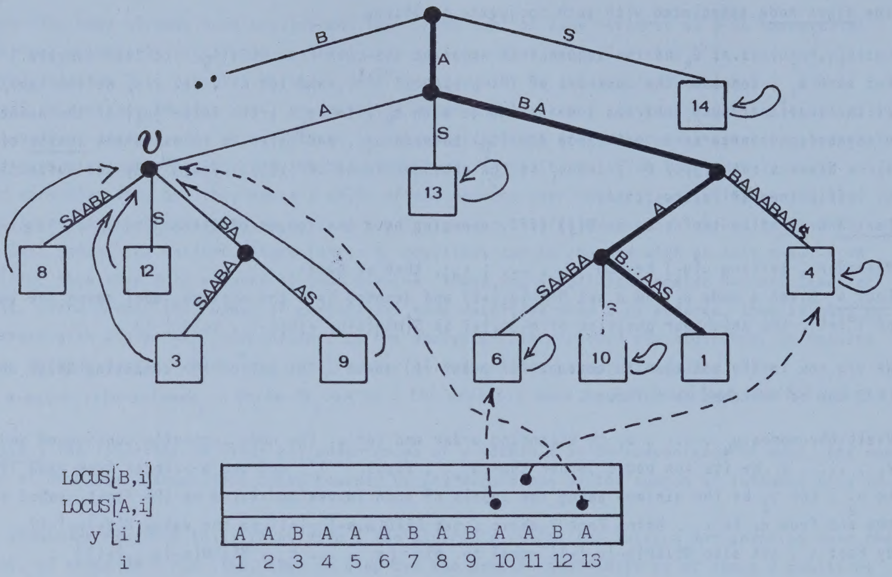


Fig. 5

Some salient features of preprocessing output . The path to leaf 1 in T_y is shown solid : only nodes on this path and their son nodes need to be accessed during execution of PM .

5. Concluding Remarks

We have proved that , by elaborating on the Boyer-Moore-Galil string searching methods, it is possible to set up a 6n comparison worst case algorithm . As stated earlier , the strategy underlying our algorithm takes advantage of knowledge about the structure of the pattern that is trascured by other methods .It must be remarked that this information about the pattern is collected by means of algorithms linear in the pattern lenght.

Due to the large preprocessing overhead, the motivation of our approach seems more combinatorial than applicative. However, there are special cases that further enhance the performance of the algorithm : for instance, assuming that the sets C(j)/a contain each at most one element , then the algorithm proposed here shrinks so that the 6n bound tightens to 3n . Such performance might motivate practical applications too.

Finally the discussion of the preceding section seems to suggest that there might be still room for improving either preprocessing , or string searching strategies , or both .

References

1 D.E. Knuth , J.H. Morris and V.B. Pratt , Fast Pattern Matching in Strings , SIAM J. on Computing 6 (1977) , 189-195 .

2 A.V. Aho and M.J. Corasick , Efficient String Matching : An Aid to Bibliografic Search , Comm. ACM 18 (1975) , 333-340 .

3 R.S. Boyer and J.S. Moore , A Fast String Searching Algorithm , Comm. ACM 20 (1977) , 262-272 .

4 T.A. Bailey and R.G. Dromey , Fast String Searching by Finding Subkeys in Subtext , Information Processing Letters 11 (1980) , 130-133 .

5 A.C.C. Yao , The Complexity of Pattern Matching for a Random String , manuscript , Computer Science Department , Stanford University , Stanford , Calif. (1977) .

6 Z. Galil , On Improving the Worst Case Running Time of the Boyer-Moore String Searching Algorithm , Comm. ACM 22 (1979) , 505-508 .

7 L.J. Guibas and A.M. Odlyzko , A New Proof of the Linearity of the Boyer-Moore String Searching Algorithm , Proc. 18th Annual IEEE Symposium on Fundation of Computer Science (1977), 189-195 .

8 E.M. McCreight , A Space economical Suffix Tree Construction Algorithm , Journal of ACM 23 (1976) , 262-272 .

THE N-TH POWER OF A COMPANION MATRIX: FAST SOLUTIONS TO LINEAR RECURRENCES

CHARLES M. FIDUCCIA
General Electric R+D Center
Schenectady, New York 12345

ABSTRACT

The solution to a linear recurrence can be expressed in terms of the vector of initial values multiplied by a power of a companion matrix. We show how to compute this matrix power efficiently by performing modular polynomial multiplication rather than matrix multiplication. As a result we obtain a simple, efficient, closed-form expression for the solution.

INTRODUCTION

Let the infinite sequence $F_0, F_1, F_2, \ldots$ be a solution to the k-th order linear recurrence

$$F_{n+k} = c_0 F_n + c_1 F_{n+1} + \ldots + c_{k-1} F_{n+k-1} \tag{1}$$

We are interested in efficiently computing an arbitrary F_n without computing all F_i which precede it. Toward this end, let $F_{i,j}$ denote the row vector $(F_i, \ldots, F_j)$. Equation (1) can now be written in matrix form

$$F_{n+1,n+k} = F_{n,n+k-1} \cdot C \tag{2}$$

where C is the k x k companion matrix

$$C = \begin{bmatrix} 0 & & & & c_0 \\ 1 & & & & c_1 \\ & 1 & & & c_2 \\ & & \cdot & & \cdot \\ & & & \cdot & \cdot \\ & & & \cdot & \cdot \\ & & & 1 & c_{k-1} \end{bmatrix}$$

all of whose entries are zero, except the 2k-1 shown. From (2) we derive

$$F_{i+n,i+n+k-1} = F_{i,i+k-1} \cdot C^n$$

so that, in terms of the vector of initial values $F_{0,k-1}$, we have

$$F_{n,n+k-1} = F_{0,k-1} \cdot C^n \tag{3}$$

Thus to compute the n-th term F_n of (1) we need only compute the n-th power of the matrix C and then multiply its first column by the vector of initial values. Because this last step is easy, we may confine our attention to the problem of computing C^n.

In any semigroup, the n-th power x^n of any element x can be computed with at most 2logn semigroup multiplications by the well-known method of repeated squaring:

$$x^{2n} = (x^n)^2 \qquad x^{2n+1} = x(x^n)^2 \qquad x^1 = x$$

Urbanek [7] suggests this approach for computing F_n with $O(k^3 \log n)$ arithmetic operations, using $O(k^3)$ operations per multiplication of k x k matrices. Gries and Levin [6] give more efficient recursive formulas for computing the required entries of C^n with $O(k^2 \log n)$ operations; however, their method sheds no light on what these entries are.

We show that, for the purpose of computing C^n, matrix multiplication can be replaced by modular polynomial multiplication. This not only reveals what the entries are, but also yields a simple, efficient, closed-form expression for F_n that immediately explains the result of Gries and Levin and opens up the possibility of an O(k logk logn) algorithm.

ARITHMETIC WITH C

In the sequel, the underlying scalar domain is assumed to be an arbitrary commutative ring K with 1. K[X] is the ring of polynomials over K and K[X]/(f(X)) is the ring of polynomials modulo the monic polynomial

$$f(X) = X^k - (c_0 + c_1X + \dots + c_{k-1}X^{k-1}) \tag{4}$$

For concreteness, we view K[X]/(f(X)) as the set of all polynomials, in K[X], of degree less than k, with arithmetic being done modulo f(X). The congruence class of K[X]/(f(X)) containing X will be denoted by ξ. This means that for any p(X) in K[X] its remainder mod f(X) is $p(\xi)$. Note that in particular $f(\xi) = 0$; so that

$$\xi^k = c_0 + c_1\xi + \dots + c_{k-1}\xi^{k-1}$$

It is well known that f(X) is the characteristic polynomial of the companion matrix C; so that f(C) = 0. This shows that one way of computing C^n is to first compute $r(\xi) = \xi^n$, by the method of repeated squaring, and then computing $C^n = r(C)$.

The above arguments reduce the problem to doing fast multiplication in the ring K[X]/(f(X)). This can always be done naively with $O(k^2)$ arithmetic operations because every element in this ring "is" a polynomial of degree less than k in K[X]. Multiplying two such elements, in K[X], uses $O(k^2)$ operations, as does the long division by f(X) to obtain the desired remainder $r(\xi)$ in K[X]/(f(X)). In fact, modular multiplication can often be done much faster [2,3]. If K supports a Fast fourier transform, then

$O(k \log k)$ operations will suffice, while $O(k \log^2 k)$ will suffice in many domains. To account for this variability, we will let $\mu(k)$ denote the complexity of doing multiplication in $K[X]/(f(X))$.

Having thus computed $r(\xi) = \xi^n$, evaluating it at $\xi = C$ can be done with an additional $O(k^3)$ operations by using Horner's rule and the fact that, owing to its sparseness, C can be multiplied by any $k \times k$ matrix with $O(k^2)$ operations. This yields an $O(k^3 + \mu(k)\log n)$ algorithm for C^n.

The above method for computing C^n is still too general and too expensive. Indeed, it works for any $k \times k$ matrix A, except for the added cost of computing the characteristic polynomial of A, which comes free with C, and a more expensive $O(k^4)$ computation of $r(A)$. Both of these costs are independent of n; however, they may prove expensive in practice. We show later how to compute C^n with $O(k^2 + \mu(k)\log n)$ operations.

The basis for a cleaner, more efficient solution is a simple, yet useful, result from Fiduccia [1]. Consider the identification between $K[X]/(f(X))$ and K^k obtained by equating $p(\xi) = p_0 + p_1\xi + \ldots + p_{k-1}\xi^{k-1}$ with its column vector of coefficients $\tilde{p} = (p_0, p_1, \ldots, p_{k-1})$. The question that arises is which polynomial in $K[X]/(f(X))$ equals $C\tilde{p}$?

LEMMA For any $p(\xi)$ in $K[X]/(f(X))$, $C\tilde{p} = \xi p(\xi)$.

Proof By trivial, direct computations of $C\tilde{p}$ and $\xi p(\xi) = Xp(X) \bmod f(X)$. The latter can be done by a simple, one-step, long division or the use of $f(\xi) = 0$. More elegantly, and a hint of things to come, note that the i-th column of C is ξ^i so that $C\tilde{p} = p_0\xi + \ldots + p_{k-1}\xi^k = \xi p(\xi)$.

COROLLARY 1

For any $q(X)$ in $K[X]$ and any $p(\xi)$ in $K[X]/(f(X))$, $q(C)\tilde{p} = q(\xi)p(\xi)$.

Proof By induction on the degree of $q(X)$. Write $q(X) = r(X)X + q_0$; so that $q(C)\tilde{p} = r(C)(C\tilde{p}) + q_0\tilde{p}$. Using the Lemma and the inductive hypothesis, this is equal to $r(\xi)(\xi p(\xi)) + q_0 p(\xi) = (r(\xi)\xi + q_0)p(\xi) = q(\xi)p(\xi)$.

The observant reader will have noticed that we are working on the wrong problem! If we take $q(X) = X^n$ in Corollary 1, we will compute $C^n\tilde{p}$ rather than $\tilde{p}C^n$ as required by (3). We resolve this problem by using Corollary 1 to compute the required column of C^n rather than $\tilde{p}C^n$ itself.

COROLLARY 2

For any $q(X)$ in $K[X]$, the i-th column of $q(C)$ is $q(\xi)\xi^{i-1}$. In particular, the i-th column of C^n is ξ^{n+i-1} and its first column is ξ^n.

Proof Choose $p(\xi) = \xi^{i-1}$ in Corollary 1, so that $\tilde{p}$ is the i-th column of the k x k identity matrix; i.e., consider the identity $q(C) = q(C)I$. Since the i-th column of I is ξ^{i-1}, the i-th column of q(C), on the left, is $q(C)\tilde{p} = q(\xi)\xi^{i-1}$. Take $q(X) = X^n$ to prove the second part; then let $i = 1$ to prove the third part.

COROLLARY 3
If $q(\xi)$ can be computed with N arithmetic operations, then q(C) can be computed with $N + k^2$ operations. In particular, C^n can be computed with $O(k^2 + \mu(k)\log n)$ operations.

Proof By Corollary 2 and the hypothesis, the first column $q(\xi)$ of q(C) can be computed with N operations. Using Corollary 2 again, along with the Lemma, the (i+1)-th column of q(C) is C times the i-th column. Thus, once column 1 is available, we can compute the others by multiplying the most current one by C. Each of these k-1 multiplications by C takes at most 2k operations, for a total of $O(k^2 + N)$. Since ξ^n can be computed with $N = O(\mu(k)\log n)$ operations, the total for C^n is thus $O(k^2 + \mu(k)\log n)$.

AN EXPRESSION FOR F_n

From (3) we see that F_n is nothing more than the row vector $F_{0,k-1}$ of initial values times the first column ξ^n of C^n. This means that if $\gamma_0, \ldots, \gamma_{k-1}$ are the coefficients of ξ^n, then $F_n = \gamma_0 F_0 + \ldots + \gamma_{k-1}F_{k-1}$. Using the inner product operator $\langle\cdot,\cdot\rangle$ we get the closed-form expression:

THEOREM 1

$$F_n = \langle (F_0, \ldots, F_{k-1}), \xi^n \rangle \tag{5}$$

Since the inner product computation uses only O(k) arithmetic operations, the bulk of the algorithm is the computation of $\xi^n = X^n \bmod f(X)$ in the ring $K[X]/(f(X))$ of polynomials modulo the monic polynomial f(X). We may of course use any method at our disposal to compute ξ^n, making use of any special knowledge about f(X) and K itself. The reader is no doubt aware of solutions to (1) based on the roots of f(X). Those solutions are simply the expression given by Theorem 1 after an appropriate change of basis for $K[X]/(f(X))$ based on the factorization properties of f(X) over K or its extensions.

Basically, we have reduced the problem to doing efficient multiplication in the ring $K[X]/(f(X))$. This is a fundamental problem in the area of algebraic complexity of computations [1,4,5]. The fast algorithms, mentioned above, for doing multiplication in $K[X]/(f(X))$ are only assymptotically good, being based on doing polynomial reciprocals quickly. Thus in practice one may want to preprocess f(X) in order to speed-up the arithmetic in $K[X]/(f(X))$. It is worth pointing out that this is precisely what "fast transforms" are all about.

EXTENSIONS

Let us delve further into other possible solutions of the linear recurrence (1) by noting that its right-hand side is nothing more than a K-linear combination that makes sense over any K-module M (any set closed under K-linear combination). Thus, given initial values $F_0, F_1, \ldots, F_{k-1}$ in M, the recurrence will generate an infinite sequence $F_0, F_1, F_2, \ldots$ in M. This suggests that there exits a more general version of Theorem 1. The following result is its abstract version.

THEOREM 2
For any K-linear mapping L from K[X]/(f(X)) into any K-module M, the sequence $L(1), L(\xi), L(\xi^2), L(\xi^3), \ldots$ is a solution to (1) in M.

Proof It is clear from $f(\xi) = 0$ that the infinite sequence of powers of ξ

$$1, \xi, \xi^2, \xi^3, \ldots \tag{6}$$

is a solution to (1) in the ring K[X]/(f(X)). So that for all n, we have

$$\xi^{n+k} = c_0\xi^n + c_1\xi^{n+1} + \ldots + c_{k-1}\xi^{n+k-1}$$

Apply L to both sides and invoke its K-linearity to complete the proof.

If particular initial values are specified, we need only choose L such that $L(1) = F_0, \ldots, L(\xi^{k-1}) = F_{k-1}$. This can always be done because $1, \xi, \xi^2, \ldots, \xi^{k-1}$ form a basis for K[X]/(f(X)). For any element $p(\xi) = p_0 + \ldots + p_{k-1}\xi^{k-1}$ we then have $L(p(\xi)) = p_0F_0 + \ldots + p_{k-1}F_{k-1}$. Consequently, the n-th term F_n of the solution in M is again

$$F_n = \gamma_0F_0 + \gamma_1F_1 + \ldots + \gamma_{k-1}F_{k-1} \tag{7}$$

where $\xi^n = \gamma_0 + \gamma_1\xi + \ldots + \gamma_{k-1}\xi^{k-1}$. This expression is identical to (5), except that the F_i are now arbitrarily chosen initial values from any K-module M rather than the base ring K.

Theorem 2 shows that, in some sense, sequence (6) is "the universal solution" to (1), since all other solutions in all other K-modules are linear images of it. It is interesting to pursue further the reason for its "universality" in order to get an intuitive feeling for it. For notational simplicity, let's confine our attention to solutions in K and consider the infinite family given by (3) for n = 0, k, 2k, 3k, ... We may write this family as a single, infinite matrix equation over K

$$F_{0,\infty} = F_{0,k-1} \cdot C^* \tag{8}$$

where C^* denotes the k x ∞ matrix

$$C^* = (I \quad C^k \quad C^{2k} \quad C^{3k} \quad \ldots)$$

From (8) we see that $F_{0,\infty}$ is the K-linear combination

$$F_0\rho_0 + \ldots + F_{k-1}\rho_{k-1}$$

where ρ_i is the (i+1)-th infinite row of C^*. Each ρ_i might be called "a basic solution" to (1) in K since, as is evident from (8), it is the solution when the vector of initial values is chosen to be the (i+1)-th row of the k x k identity matrix. Thus the rows of C^* are precisely the k "basic" solutions from which all solutions in K - indeed in all K-modules - can be obtained. The "universality" of solution (6) in K[X]/(f(X)) is now evident from (8) and Corollary 2, since the powers of ξ are the columns of C^*

$$C^* = (1 \quad \xi \quad \xi^2 \quad \xi^3 \quad \ldots)$$

Consequently, as we generate sequence (6) we are in fact simultaneously computing all k "basic" solutions to (1) in K. This means that the i-th entry of the k-vector ξ^n is the n-th term of the i-th "basic" solution. A rather nice turn of events!

As an interesting application of these observations, consider the shift-register implementations of (1) shown below. The first shows the direct implementation of (1) over any K-module M. The boxes are delay elements, to store the current state, whose internal labels indicate the value of the initial state. At each step, the circuit performs k scalar multiplications and k-1 additions in M to compute the next term of the specific solution generated by the given initial values $F_0, \ldots, F_{k-1}$.

The second circuit, based on the Lemma and Theorem 2, generates the infinite sequence of powers $1, \xi, \xi^2, \xi^3, \ldots$ in K[X]/(f(X)) from the initial state $\xi^0 = (1, 0, 0, \ldots, 0) = 1$. The interesting thing about this circuit is that it uses only k multiplications and k-1 additions, in the base ring K, to compute all k "basic" solutions simultaneously. It does this independently of the initial values $F_0, \ldots, F_{k-1}$. These values are introduced, at any time, by performing a K-linear transformation on the current state ξ^n of the shift-register. The transformation shown is for the case M = K, assuming the standard basis $1, \xi, \xi^2, \ldots, \xi^{k-1}$ for K[X]/(f(X)); however, (7) shows that the circuit is also valid over any K-module.

One potentially practical benefit of introducing the initial values of the recurrence at the end of the computation is that at any stage, having done the bulk of the computation, one can experiment with various initial values without having to restart the computation. Alternatively, we may "fan-out" the state vector to as many linear transformations as desired and simultaneously compute as many solutions, in as many different K-modules, as desired.

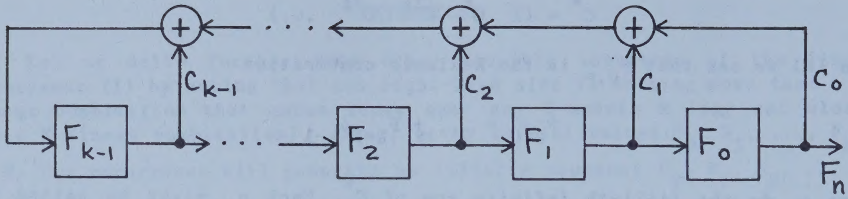


Figure 1. The obvious implementation of the recurrence (1).

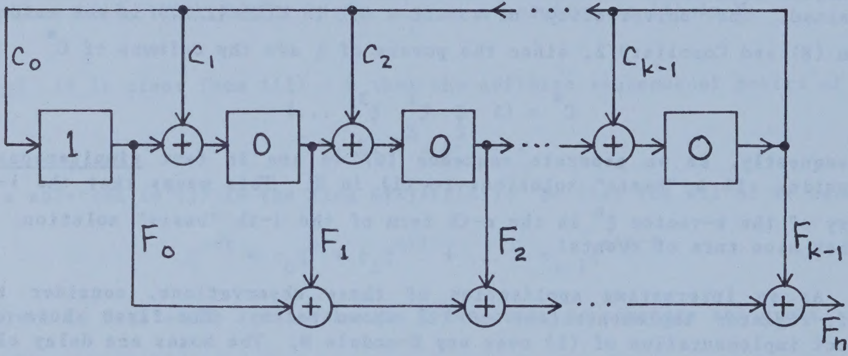


Figure 2. The implementation of (1) according to Theorems 1 or 2.

REFERENCES

[1] Fiduccia, C.M., On the algebraic complexity of matrix multiplication, Ph.D. Thesis, Brown University, Providence, R.I., 1973.

[2] Aho, A.V., J.E. Hopcroft and J.D. Ullman, The design and analysis of computer algorithms, Addison-Wesley, Reading, MA, 1974.

[3] Borodin, A. and I. Monroe, The complexity of algebraic and numeric problems, Elsevier, New York, 1975.

[4] Fiduccia, C.M. and Y. Zalcstein, Algebras having linear multiplicative complexities, J. ACM, Vol. 24, No. 2, April 1977, pp. 311-331.

[5] Winograd, S., Arithmetic complexity of computations, SIAM monograph, Philadelphia, PA, 1980.

[6] Gries,D. and G. Levin, Computing Fibonacci numbers (and similarly defined functions) in log time, Information Processing Letters, Vol. 11, No. 2, October 1980, pp. 68-69.

[7] Urbanek, F.J., An O(log n) algorithm for computing the n-th element of the solution of a difference equation, Information Processing Letters, Vol. 11, No. 2, October 1980, pp. 66-67.

HYBRID ALGORITHMS FOR NP-HARD PROBLEMS

DONALD K. FRIESEN
Texas A&M University
College Station, Texas

MICHAEL A. LANGSTON
Washington State University
Pullman, Washington

Abstract

This paper discusses hybrid algorithms for obtaining near optimal solutions to combinatorial problems. The analysis of such an algorithm for bin packing is described.

Approximation algorithms are frequently used for NP-hard problems since the time to obtain optimal solutions is excessive. For many such problems there are several available algorithms which work well in different situations. An obvious way of trying to obtain improved solutions is to run several algorithms, choosing the best of the resulting solutions. The purpose of this paper is to examine the use and analysis of such hybrid schemes.

Although the concept is simple, knowing which algorithms to combine may not be obvious. Moreover the analysis of such hybrid algorithms is difficult and few analyses have been done. In this paper we show one problem where the worst-case bound of such a hybrid method is provably better than either of the individual algorithms and discuss some other problems where the approach might be fruitful.

The bin packing problem is well known and until recently, the best bound known for a polynomial time algorithm was (asymptotically) 11/9, achieved by the "first fit decreasing" algorithm (FFD) [4]. Tighter asymptotic bounds have been achieved recently in [1], [3], and [5] showing that there is a fully polynomial algorithm with an asymptotic worst-case ratio of one. The fully polynomial algorithm is not practical, however. Another algorithm, similar to FFD, is described and analyzed in [2]. In this algorithm, called the "best two fit" algorithm (B2F), after a bin has been packed in the way the FFD algorithm would pack it, an attempt is made to replace the smallest piece packed in it by two pieces. The resulting algorithm has a somewhat larger worst-case bound (5/4) than FFD; however, on the examples where the FFD algorithm approaches its worst-case performance the B2F algorithm does quite well and vice versa.

Thus it seems worthwhile to analyze the behavior of a hybrid scheme using both of these algorithms. We can prove that the resulting algorithm has an asymptotic worst-case bound lower than 6/5. The worst example we know uses about 1.164 times as many bins as the optimal number.

The proof consists of showing that certain types of packings can be ruled out if both algorithms perform as poorly as 6/5 times the optimal algorithm. For the remaining cases, we show that the average of the number of bins used by the two algorithms is less than 6/5.

REFERENCES

[1] W. Fernandez de la Vega and G. S. Lueker, "Bin packing can be solved with 1+ε in linear time," Combinatorica 1, (1981), 349-355.

[2] D. K. Friesen, "A new bin packing algorithm and its application to scheduling," Texas A&M University technical report TAMUDCS-82-008-R.

[3] M. R. Garey and D. S. Johnson, "A 71/60 algorithm for one-dimensional bin packing," unpublished manuscript (1980).

[4] D. S. Johnson, "Near-Optimal bin packing algorithms," Doctoral dissertation, Massachusetts Institute of Technology, 1973.

[5] R. M. Karp and N. Karmarker, "An efficient approximation scheme for the one-dimensional bin packing problem," Proceedings 23rd Ann. Symp. on Foundations of Computer Science, IEEE Computer Society, Long Beach, Calif. 1982 (to appear)

SEARCHING IN POLYNOMIAL AVERAGE TIME

PAUL PURDOM
Indiana University
Bloomington, Indiana

CYNTHIA BROWN
Indiana University and GTE Labs
Waltham, Mass.

Many important and interesting problem sets are NP complete. For such problem sets every algorithm that has been analyzed has a worst-case time that is an exponential function of the problem size. Recent analyses, however, show that many classes of random problems can be solved in polynomial average time, using simple search algorithms. These results are summarized in the Figure. Each point in the Figure represents a statistical model of random conjunctive normal form predicates with v variables ($2v$ literals), where a random clause is formed by independently selecting each literal with probability $p(v)$, and there are $t(v)$ clauses in each predicate. The vertical axis is for various functions for $p(v)$: small functions (as v goes to infinity) are at the bottom while large functions are at the top. The horizontal axis is for $t(v)$ with small functions at the left. For each search algorithm and each function for $p(v)$ and for $t(v)$ one can compute the asymptotic average running time. The contours in the Figure show the division between points where the average running time is known to be polynomial and where it may be larger.

Each contour in the Figure is labelled with the associated algorithm. The contour is marked with hash marks when it is the result of an upper bound analysis. The result of an exact analysis might be anywhere on the hashed side of the line. Problem sets in the region marked *hard* appear to require exponential average time with current algorithms, but since most of the boundaries of the hard region result from upper bound analyses, more careful analyses might shrink the region.

Problem sets outside the hard region are definitely easy for some algorithm. The lower backtracking curve and the upper dashed curve separate problems which, on the average, have a polynomial number of solutions from those that have an exponential number. The upper right part of the Figure shows how successively better analyses of the pure literal rule have been shrinking the hard region.

To summarize the results, let n be an arbitrary large number and ϵ an arbitrary small one. An algorithm based on the pure literal rule gives polynomial average time when (1)$p(v) \geq \epsilon$ [1], (2) $t(v) \leq n \ln v$ [5], (3) $t(v) \leq \frac{n}{p(v)}\sqrt{\frac{\ln v}{v}}$ for $p(v)$ small [5], or (4) $t(v) \leq \frac{\ln 2}{2p(v)} v^{np(v)}$ [5].

Backtracking has polynomial average time [3] when (1) $p(v) \leq 1/v$ and $t(v) \geq \frac{(\ln 2+\epsilon)v}{-\ln(1-e^{vp(v)})}$ or (2) $p(v) \geq 1/v$ and $t(v) \geq \frac{(\ln 2+\epsilon)dv}{vp(v)}$, where d is the largest root of $\ln(1+ d)+ d\ln(1+ 1/d) = vp(v)$.

We also analyzed the performance of search rearrangement backtracking [2]. It is not shown on Figure 1. We showed that this algorithm can be a significant improvement over ordinary backtracking only when the average number of solutions per problem is small (not exponential) and yet backtracking requires exponential time. This happens only at the upper right part of Figure 1 in the triangle formed by the dashed line, the backtracking line, and the pure literal line. We showed that in this region the contour for search rearrangement backtracking is slightly to the left of that for ordinary backtracking. Most of the region between the dashed line and the backtracking line still requires exponential average time even when search rearrangement backtracking is used.

Other analyses shown on the Figure are given in [3,4].

References

[1] Allen Goldberg, Paul Walton Purdom Jr., and Cynthia A. Brown, *Average Time Analysis of Simplified Putnam-Davis Procedures*, Information Processing Letters (to appear).

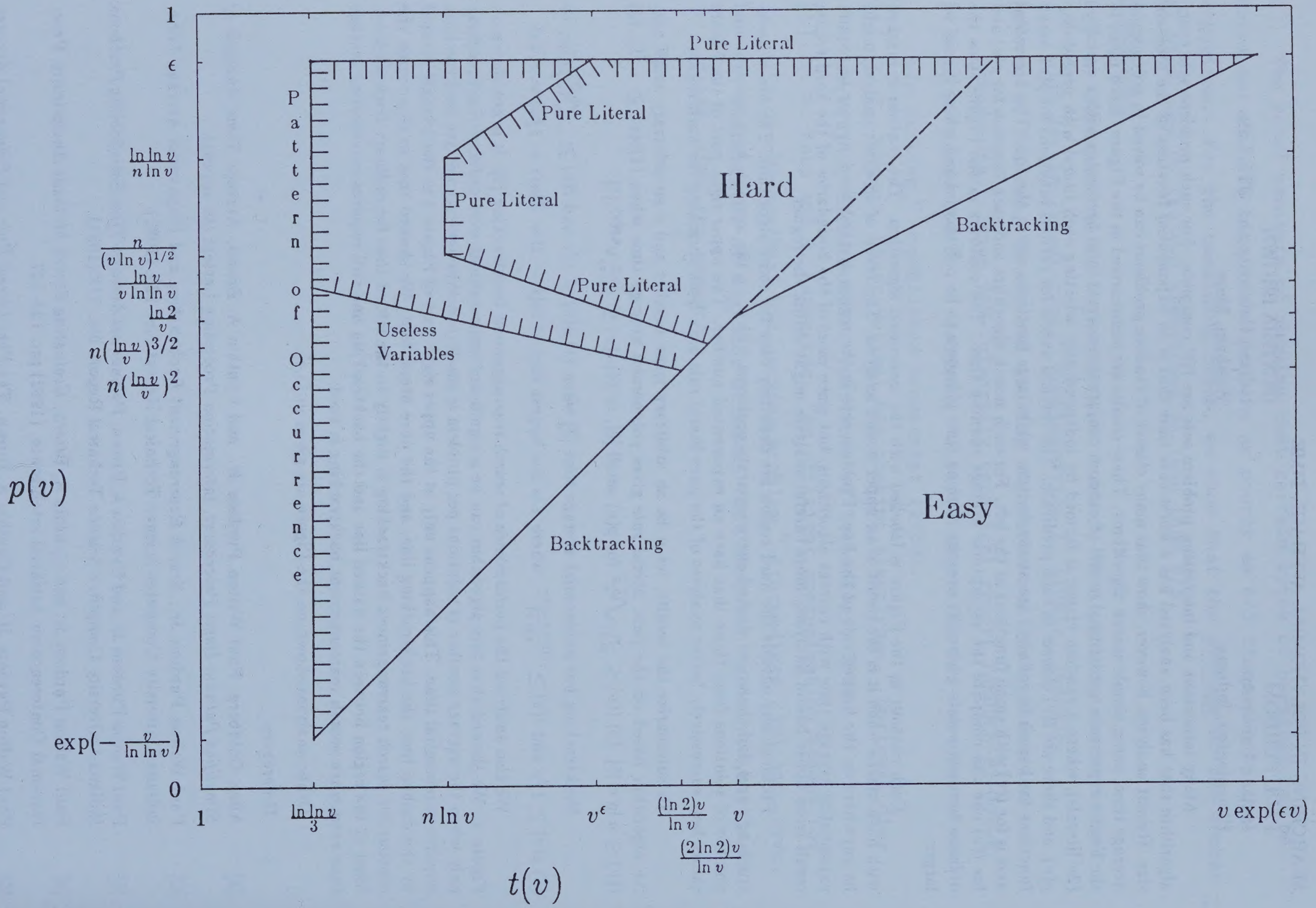
[2] Paul Walton Purdom Jr., *Search Rearrangement Backtracking and Polynomial Average Time*, Indiana University Computer Science Technical Report No. 123, (1982).

[3] Paul Walton Purdom Jr. and Cynthia A. Brown, *Polynomial Average-Time Satisfiability Problems*, Indiana University Computer Science Technical Report No. 118 (1981).

[4] Paul Walton Purdom Jr. and Cynthia A. Brown, *Evaluating Search Methods Analytically*, Proc. National Conference on Artificial Intelligence, (1982) pp. 124–127.

[5] Paul Walton Purdom Jr. and Cynthia A. Brown, *The Pure Literal Rule and Polynomial Average Time*, Indiana University Computer Science Technical Report No. 128 (1982).

Research reported herein was supported in part by NSF grant MCS 7906110.



1
ϵ
$\frac{\ln\ln v}{n\ln v}$
$\frac{n}{(v\ln v)^{1/2}}$
$\frac{\ln v}{v\ln\ln v}$
$\frac{\ln 2}{v}$
$n(\frac{\ln v}{v})^{3/2}$
$n(\frac{\ln v}{v})^{2}$
$\exp(-\frac{v}{\ln\ln v})$
0
$p(v)$
Pure Literal
Pure Literal
Pure Literal
Pure Literal
Hard
Backtracking
Pattern of Occurrence
Useless Variables
Easy
Backtracking
1
$\frac{\ln\ln v}{3}$
$n\ln v$
v^{ϵ}
$\frac{(\ln 2)v}{\ln v}$
v
$\frac{(2\ln 2)v}{\ln v}$
$v\exp(\epsilon v)$
$t(v)$

MINIMAX ROBUST EQUALIZATION FOR RANDOM SIGNALS THROUGH UNCERTAIN CHANNELS†

GEORGE MOUSTAKIDES
Dept. of Elec. Eng. & Comp. Sci.
Princeton University
Princeton, N.J. 08540

SALEEM A. KASSAM
Dept. of Systems Eng.
University of Pennsylvania
Philadelphia, PA 19104

ABSTRACT

The problem of fixed filter design for equalization of an uncertain channel is considered. A robust filter is derived by seeking the minimax filter for channel characteristics defined by bounds on amplitude and phase functions for the mean-squared-error performance criterion. An application of the results in equalization of a multipath channel is considered, and numerical values are given for the performance of the robust filter.

I. INTRODUCTION

The use of some form of channel equalization is mandatory in many transmission schemes where the channel characteristics cannot be assumed to be ideal, that is, where linear amplitude and phase distortion is present. In many situations the channel characteristics are not simply non-ideal but are in general not exactly known and may be time-varying. One approach which has been widely applied in equalization in such cases is that of adaptive schemes. In some cases, however, the use of adapative equalizers may be impractical because of cost and complexity as well as speed-of-convergence requirements.

In this paper we apply the minimax formulation to the problem of fixed equalizer design for uncertain channel characteristics. Let $G(\omega)$ be the frequency response of the linear channel, and assume that a random signal with power spectral density $S(\omega)$ is transmitted through the channel and observed in additive noise with power spectrum $N(\omega)$ at the output of the channel. The problem is to design an equalizer with some frequency response $H(\omega)$ such that the output is a good estimate, in the mean-squared-error sense, of the transmitted signal. In the minimax approach we define a class of possible channel characteristics, and seek the equalizer $H_R(\omega)$ which minimizes its worst-case performance, in terms of mean-squared-error, over the class of channel characteristics. The equalizer $H_R(\omega)$ is thus the minimax (robust) equalizer.

We consider the specific class of channel characteristics defined by upper and lower bounding functions for the amplitude characteristic $|G(\omega)|$ and with the phase characteristics $\arg G(\omega)$ lying in subsets $\Phi(\omega)$ of $[-\pi,\pi]$ (as a function of ω). The solution $H_R(\omega)$ for this particular class is obtained explicitly, and is shown to have a very interesting interpretation. At frequencies for which the signal-to-noise is high relative to channel uncertainties, the robust equalizer simply inverts the "average" channel. Otherwise, a Wiener-filter type of equalizer is obtained. We also present a numerical example to illustrate our results. The example considers an ideal transmission path and a secondary path with inexactly known amplitude and time-delay. The signal has a first-order low-pass spectrum and the noise is white for our example.

---

† This research is supported by the Office of Naval Research under Contract N0014-80-K-0945.

It is interesting to note that the problem of fixed equalizer design for an ensemble of random channels was considered in [1], using a statistical approach rather than the minimax approach. A slightly more general cost function was utilized in [1], incorporating amplitude scaling. The results in [1] are, nonetheless, of the same type as those obtained here. Further work will include use of the amplitude-scaled cost function in our formulation, as well as a consideration of uncertainties in signal and noise spectra.

II. PROBLEM DEFINITION

The observed waveform at the channel output may be written as $(g*s)(t) + n(t)$, where g is the channel impulse response with corresponding frequency response $G(\omega)$ and $s(t)$, $n(t)$ are the input random signal and additive noise with zero means and respective power spectral densities $S(\omega)$ and $N(\omega)$. The signal and noise are assumed to be uncorrelated and their power spectra are assumed to be known. Let a filter with frequency response $H(\omega)$ be used at the channel output to obtain an estimate of the signal waveform. Then the mean-squared-error $e(G,H)$ between the signal $s(t)$ and its estimate can be easily shown to be

$$e(G,H) = \frac{1}{2\pi} \int_{-\infty}^{\infty} |1-G(\omega)H(\omega)|^2 S(\omega) + |H(\omega)|^2 N(\omega) \quad d\omega \tag{1}$$

We now assume that the channel frequency response is not precisely known, but that its amplitude characteristic is bounded by known functions A_L, A_U, so that

$$A_L(\omega) \leq |G(\omega)| \leq A_U(\omega) \tag{2}$$

Furthermore, the argument $\phi(\omega)$ of $G(\omega) = |G(\omega)|\exp[j\phi(\omega)]$ is assumed to lie in a known closed subset $\Phi(\omega)$ of $[-\pi,\pi]$ for each ω. Let the class of allowable frequency responses G so defined be $\mathcal{G}$. Our objective is to find the filter frequency response $H_R(\omega)$ which will satisfy

$$\min_{H} \max_{G \in \mathcal{G}} e(G,H) = \max_{G \in \mathcal{G}} e(G,H_R), \tag{3}$$

so that H_R will be the frequency response of a minimax robust filter which optimizes worst-case estimation performance, chosen from the class of all linear filters.

III. SOLUTION FOR MINIMAX FILTER

In (1), we first fix H and obtain max $e(G,H)$. To do this we need to maximize pointwise the term $|1 - G(\omega)H(\omega)|^2$ in the integrand in (1). Now this can be written as

$$|1-G(\omega)H(\omega)|^2 = 1 - 2|G(\omega)||H(\omega)|\cos(\phi(\omega) + \theta(\omega)) + |G(\omega)|^2 H(\omega)|^2 \tag{4}$$

where $H(\omega) = |H(\omega)|\exp[j\theta(\omega)]$.

For any value of $|G(\omega)|$, this is maximized by picking $\phi(\omega) \in \Phi(\omega)$ to minimize $\cos[\phi(\omega) + \theta(\omega)]$. This minimizing value $\phi_W(\omega)$ of $\phi(\omega)$ is given as $\arg \min_{\phi \in \Phi} |(\phi+\theta)\bmod(2\pi)-\pi|$. We now consider maximization with respect to $|G|$[1]. Note that $1-2|G||H|\cos(\phi_W + \theta) + |G|^2 |H|^2$ is quadratic in $|G|$ and has a minimum at $|G| = \cos(\phi_W+\theta)/|H|$. It is therefore clear that the maximizing value $|G|_W$ of $|G|$ is

$$|G(\omega)|_W = \begin{cases} A_L(\omega), & \cos[\phi_W(\omega) + \theta(\omega)]/|H(\omega)| \geq [A_L(\omega)+A_U(\omega)]/2 \\ A_U(\omega), & \text{otherwise} \end{cases} \tag{5}$$

We now consider the minimization with respect to H of $e(G_W,H)$, where $G_W = |G|_W\exp[j\phi_W]$. First consider minimization with respect to the phase $\theta(\omega)$. We interpret $\Phi(\omega)$ as some set of points on the unit circle in the complex plane, and let $2\alpha(\omega)$ be the largest angle subtended (in radians) at the origin by any arc on the unit circle not in $\Phi(\omega)$. Let $\beta(\omega)$ be the angular location of the middle of this arc (see Figure 1). Then the filter phase $\theta(\omega) = \theta_R(\omega)$ minimizing $e(G_W,|H|\exp[j\theta])$ is

$$\theta_R(\omega) = \pi-\beta(\omega), \tag{6}$$

or that phase which when added to $\beta(\omega)$ transforms this point to the point π on the unit circle. With this phase angle we have

$$e(G_W,|H|\exp[j\theta_R]) = \frac{1}{2\pi}\int_{-\infty}^{\infty} \{1 + 2\,|G|_W|H|\cos\alpha + |G|_W^2\,|H|^2\}S + |H|^2 N \quad d\omega \tag{7}$$

The integrand in (7) can be written as

$$I = S + (2S|G|_W\cos\alpha)|H| + (|G|_W^2\,S+N)|H|^2 \tag{8}$$

which is a quadratic in $|H|$. Note that in this expression $|G|_W = A_L$ when $|H|$ is less then or equal to $-2\cos\alpha/(A_L+A_U)$, and is A_U otherwise. Note that for $|H| = 0$ or $-2\cos\alpha/[A_L+A_U]$, I has the same value for both $|G|_W = A_L$ and A_U. It is also clear that when $\cos\alpha$ is non-negative, the minimizing value

---

[1] The argument ω is dropped henceforth wherever clarity is not affected.

$|H| = |H|_R$ is $|H_R| = 0$. We therefore consider the case where $\cos\alpha < 0$. Now if we put $|G|_w = A_U$ in (8), the resulting function $I_U(|H|)$ has a minimum at a value of $|H|$ less than or equal to $-2\cos\alpha/(A_L+A_U)$, at which point I of (8) has $|G|_w = A_L$. This follows from the fact that the minimum for $I_U(|H|)$ occurs at $|H|^* = -SA_U\cos\alpha/(A_U^2 S + N)$ so that $1/|H|^* \geq -A_U/\cos\alpha \geq -(A_U+A_L)/2\cos\alpha$. Let $I_L(|H|)$ be I of (8) as a function of $|H|$ when $|G|_w = A_L$. Then depending on whether $I(|H|)$ is described by Figure 2(a) or 2(b), we finally get that the minimizing value $|H| = |H|_R$ of I is

$$|H_R(\omega)| = \begin{cases} 0 , & \cos[\alpha(\omega)] \geq 0 \\[2ex] \dfrac{-2\cos[\alpha(\omega)]}{A_L(\omega) + A_U(\omega)} , & \dfrac{S(\omega)}{N(\omega)} \geq \dfrac{2}{A_L(\omega)[A_U(\omega) - A_L(\omega)]} \\ & \text{and } \cos[\alpha(\omega)] < 0 \\[2ex] \dfrac{-S(\omega)A_L(\omega)\cos[\alpha(\omega)]}{A_L^2(\omega)S(\omega) + N(\omega)}, & \text{otherwise} \end{cases} \tag{9}$$

Note that (6) gives the phase of the minimax filter.

From the definition of $\alpha(\omega)$ we see that it is a measure of the degree to which channel phase is known. For $\alpha(\omega) \leq \pi/2$ the minimax equalizer produces zero output, the mean value of the desired signal, regardless of the degree to which the channel amplitude characteristic is known. Otherwise, note that the condition in (9) dictating the choice of filter amplitude is $N(\omega)/[A_L^2(\omega)S(\omega)] \leq \frac{1}{2}\{[A_U(\omega)/A_L(\omega)]-1\}$. The right side of this inequality is a measure of channel amplitude uncertainty, while the left side is the maximum noise-to-signal ratio at the channel output. Therefore when noise-to-signal ratio is not larger than the measure of channel uncertainty, the robust filter is essentially the inverse of the "average" channel. Otherwise noise suppression is more important and the filter behaves essentially as a Wiener filter for the channel with lowest gain.

IV. UNCERTAIN MULTIPATH CHANNELS - AN EXAMPLE

Consider the particular situation where the channel freuqency response is $G(\omega) = 1+a\exp(-j\omega\tau)$ where the secondary path attenuation $1/a$ and time delay τ are not precisely known. Let the noise be white with spectral density N_0 and let the signal spectrum be

$$S(\omega) = \frac{N_0}{0.1 + 0.0695\tau_0^2\omega^2} , \tag{10}$$

where N_0 and τ_0 are known parameters. We model the multipath uncertainty by $0.2 \leq a \leq 0.8$ and $\tau \leq \tau_0$. Since τ may be zero, it it clear that the upper bound $A_U(\omega)$ for $|G(\omega)|$ is 1.8. To determine $A_L(\omega)$ we observe that for ω such that $\omega\tau_0 \geq \pi$ we can always find a value $\tau \leq \tau_0$ such that $\omega\tau = \pi$ so that $A_L(\omega) = 0.2$ for $\omega \geq \pi/\tau_0$. From the geometry in Figure 3, we find that $A_L(\omega)$ is in general given by

$$A_L(\omega) = \begin{cases} [1.04 + 0.4\cos(\omega\tau_0)]^{\frac{1}{2}}, & \omega\tau_0 \leq 101.5^\circ \\ \sin(\omega\tau_0), & 101.5^\circ \leq \omega\tau_0 \leq 143.1^\circ \\ [1.64 + 1.6\cos(\omega\tau_0)]^{\frac{1}{2}}, & 143.1^\circ \leq \omega\tau_0 \leq 180^\circ \\ 0.2, & \omega\tau_0 \geq \pi \end{cases} \tag{11}$$

From the geometry in Figure 3, we can similarly derive the upper and lower bounds defining the uncertainty intervals $\Phi(\omega)$ for $\phi(\omega)$. Without utilizing the above amplitude constraints, that is, considering only extreme values of the phase $\phi(\omega)$, we obtain for $\phi_L(\omega)$ and $\phi_U(\omega)$ defining $\Phi(\omega) = [\phi_L(\omega), \phi_U(\omega)]$ the values

$$\phi_L(\omega) = \begin{cases} -\sin^{-1} \dfrac{0.8\sin(\omega\tau_0)}{[1.64 + 1.6\cos(\omega\tau_0)]^{\frac{1}{2}}}, & 0 \leq \omega\tau_0 \leq 143.1^\circ \\ -53.1^\circ, & \omega\tau_0 \geq 143.1^\circ \end{cases} \tag{12}$$

and

$$\phi_U(\omega) = \begin{cases} 0, & 0 \leq \omega\tau_0 \leq \pi \\ -\sin^{-1} \dfrac{0.8\sin(\omega\tau_0)}{[1.64 + 1.6\cos(\omega\tau_0)]^{\frac{1}{2}}}, & \pi \leq \omega\tau_0 \leq 216.9^\circ \\ 53.1^\circ, & \omega\tau_0 \geq 216.9^\circ \end{cases} \tag{13}$$

We now use the above separately derived bounds for amplitude and phase characteristics to define the channel uncertainty. Note that this process results in a somewhat larger uncertainty class than is defined by the original bounds on a and τ. The quantity $\cos[\alpha(\omega)]$ in the result (9) is negative for all ω. The point of intersection of the characteristic S/N and $2/A_L(A_L - A_U)$ is at $\omega\tau_0 \approx 2\pi/3$.

Thus we get for the minimax robust filter the characteristic

$$|H_R(\omega)| = \begin{cases} \dfrac{2\cos\{[\phi_L(\omega) + \phi_U(\omega)]/2\}}{A_L(\omega) + A_U(\omega)} , & |\omega\tau_0| \leq \dfrac{2\pi}{3} , \\ \dfrac{A_L(\omega)\cos\{[\alpha_L(\omega) + \phi_U(\omega)]/2\}}{A_L^2(\omega) + 0.1 + 0.0695(\omega\tau_0)^2} , & |\omega\tau_0| > \dfrac{2\pi}{3} , \end{cases} \tag{14}$$

where ϕ_L, ϕ_U, and A_L are given in (11)-(13) and $A_U = 1.8$. The phase of the robust filter is $\theta_R(\omega) = -[\phi_L(\omega) + \phi_U(\omega)]/2$.

Without the minimax approach one might base the equalizer filter design on some "nominal" channel characteristic. In particular, we may take here the nominal channel frequency response to be $G_N(\omega) = 1 + 0.5\exp(-j\omega\tau_0/2)$. Let the filter which is optimal for this nominal channel have frequency response $H_N(\omega)$. The amplitude characteristic of this filter is given by

$$|H_N(\omega)| = \frac{[1.25 + \cos(\omega\tau_0/2)]^{\frac{1}{2}}}{1.25 + \cos(\omega\tau_0/2) + 0.1 + 0.0695(\omega\tau_0)^2} \tag{15}$$

In Figure 4 we compare the amplitude characteristics of the robust and nominal filters.

Some explicit values of mean-squared-error obtained with the nominal and robust filters are as follows: $e(G_N,H_N) = 4.76\ (2N_0/\tau_0)$, $e(G_N,H_R) = 5.36\ (2N_0/\tau_0)$, $e(G_U,H_N) = 7.4\ (2N_0/\tau_0)$ and $e(G_U,H_R) = 6.94\ (2N_0/\tau_0)$. Here $G_U(\omega) = 1 + 0.8\exp(-j\omega\tau_0)$, i.e. the channel with the strongest secondary path and with maximum path difference. Note that the robust filter loses some performance under nominal conditions, but performs better under this deviation from nominal assumptions.

V. DISCUSSION AND CONCLUSION

The above numerical example provides an illustration of the practical application of the results of this paper. While the performance of the minimax robust filter in this case was not very much better than that of the nominal filter, it should be kept in mind that the multipath channel uncertainties involved only two unknown parameters, from which a larger class of channel frequency responses was derived. Thus the robust filter is designed for a larger uncertainty class than originally specified. In light of this, the performance of the robust filter for the "nominal" channel is quite good. It is possible to show significant gain in performance for the robust filter relative to the nominal filter for other channel characteristics chosen from this larger uncertainty class.

An important observation made in section III was that at frequencies for which the maximum noise-to-signal ratio is less than a measure of channel uncertainty, the minimax robust equalizer is essentially the inverse of the "average" channel. Here the "average" channel has frequency response which lies in the "center" of the bounded uncertainty class, and the measure of channel uncertainty is $[(A_U(\omega)/A_L(\omega))-1]/2$. Note that this measures only amplitude uncertainty. In [1] the problem of equalization of a random channel characterized by an ensemble of possible channel responses was considered. One basic conclusion there was also very similar to the above observation. For frequencies where the maximum noise-to-signal ratio is large relative to channel uncertainty noise rejection becomes important and a Wiener-filter type of behavior is indicated. At these frequencies the minimax equalizer characteristic depends on the signal and noise spectra. Thus an extension of this work should consider uncertainties for these spectra also.

REFERENCE

[1[F. J. Brophy, G.J. Foschini, and R.D. Gitlin, "A Compromize Equalizer Design Incorporating Performance Invariance", BSTJ, Vol. 52, pp. 1077-1095, Sept. 1973.

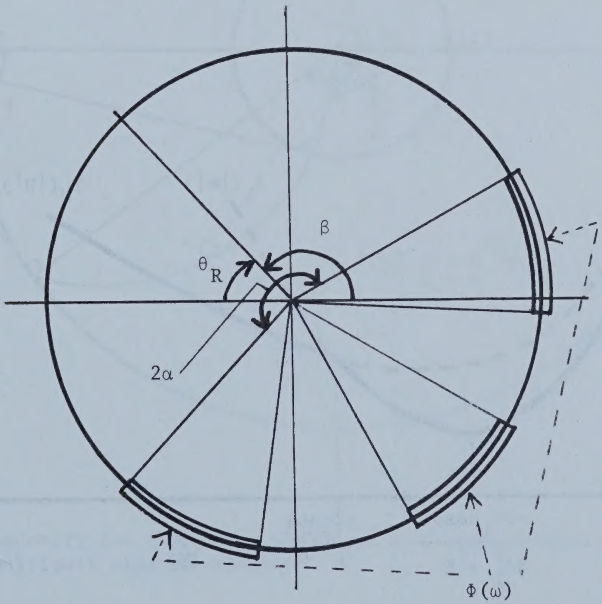


Figure 1. Illustration of the uncertainty set $\Phi(\omega)$ and the angles $\alpha(\omega)$, $\beta(\omega)$ and $\theta_R(\omega)$.

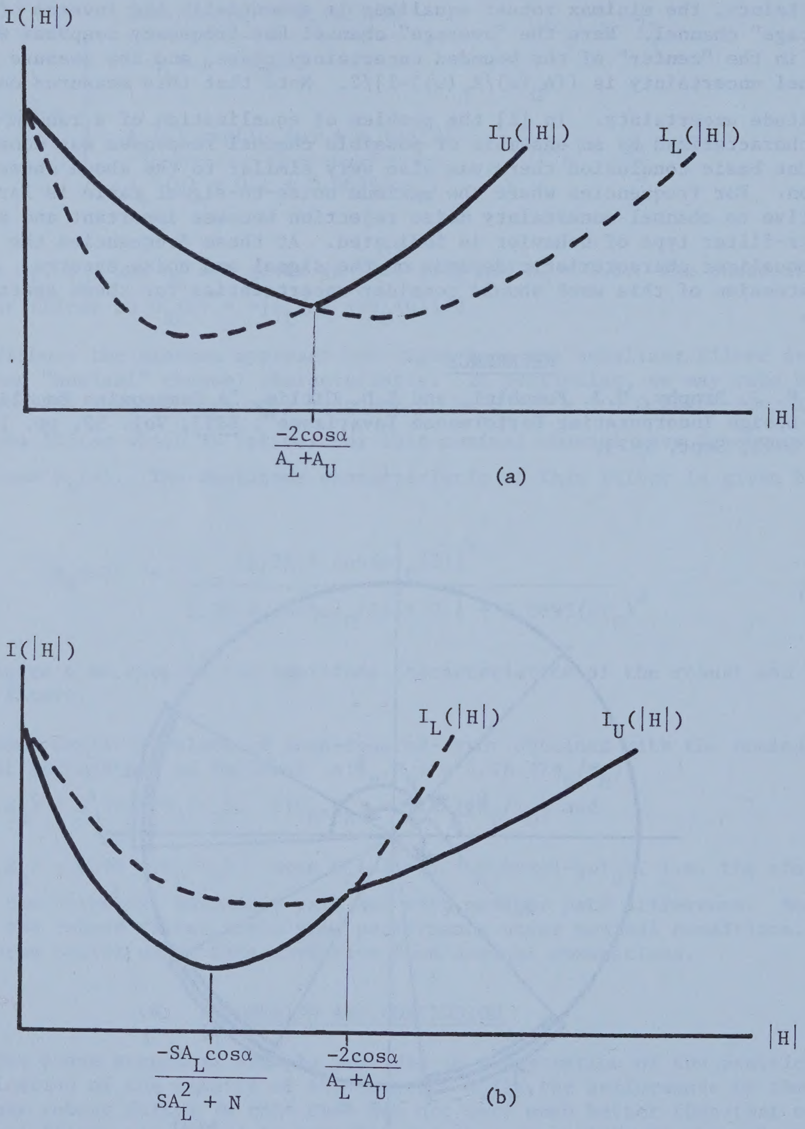


Figure 2. Minimization of $I(|H|)$ [equation (8)] for solution of robust filter magnitude.

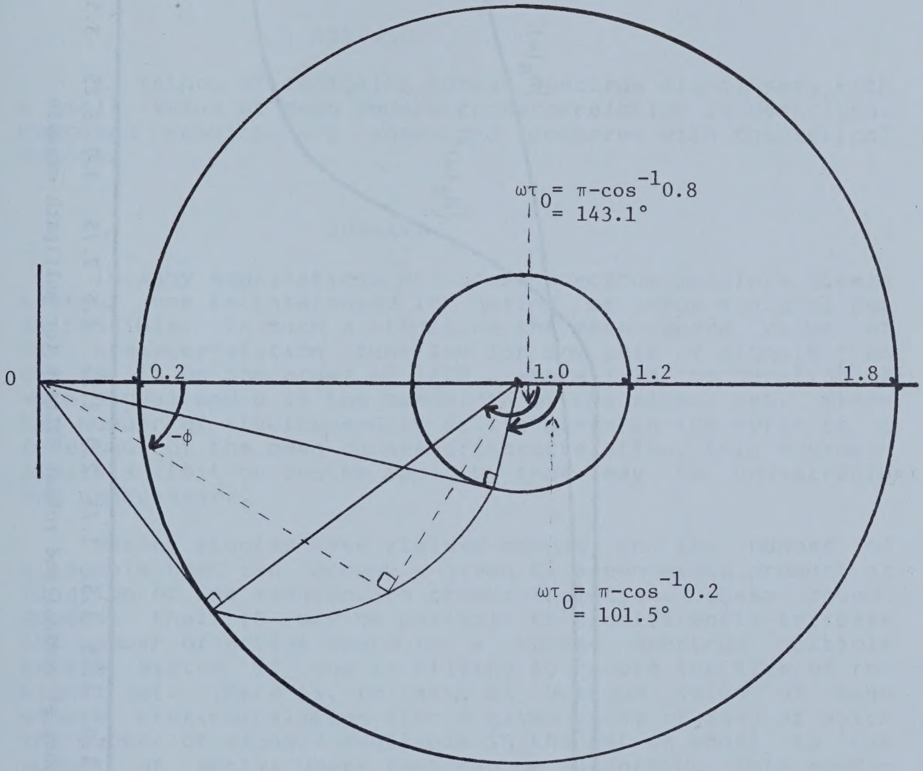

ωτ0= π-cos-1 0.8
= 143.1°
0
0.2
1.0
1.2
1.8
-φ
ωτ0= π-cos-1 0.2
= 101.5°

Figure 3. Geometry for computing amplitude and phase bounds for multipath channel example.

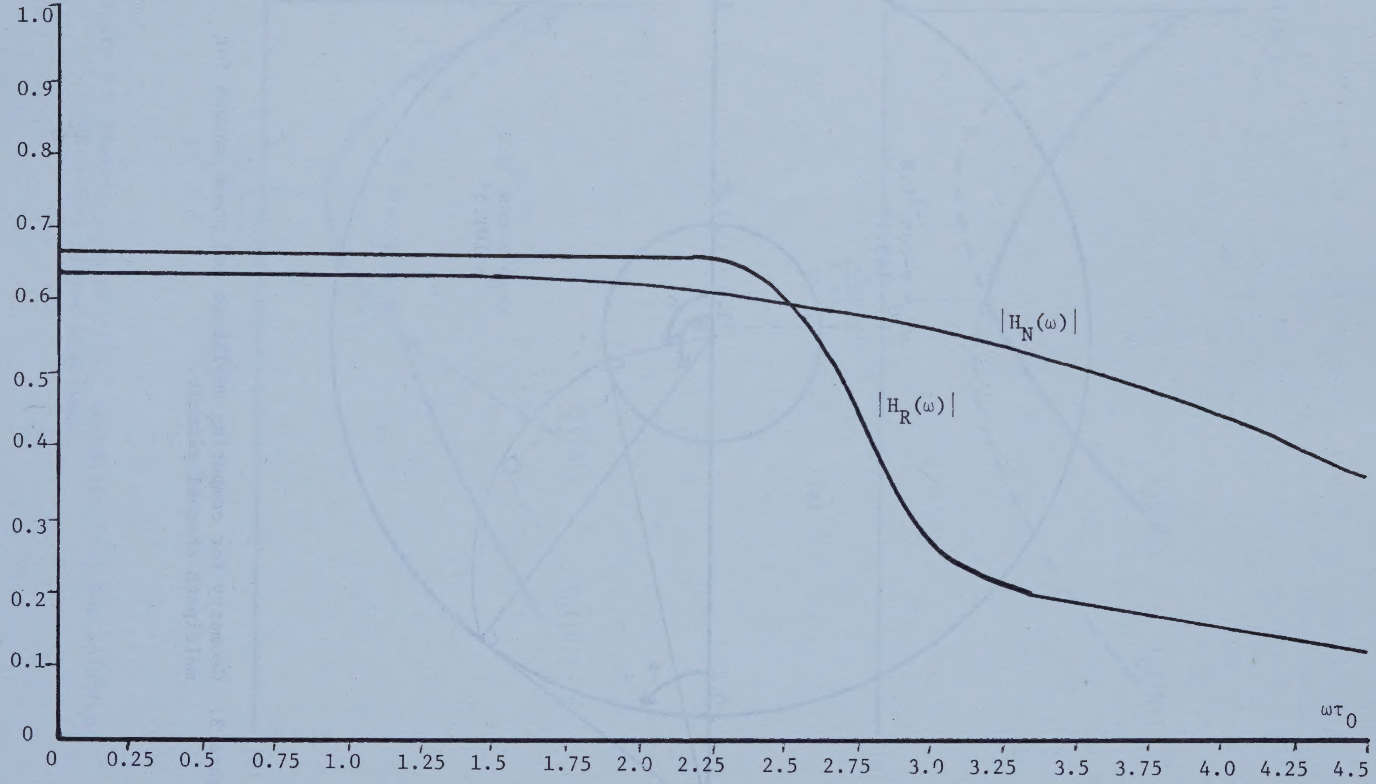


Figure 4. Magnitudes of nominal and robust filters for multipath example

DESIGN OF SPREAD SPECTRUM SIGNAL SETS WITH SMALL MEAN-SQUARE CROSSCORRELATION

G. R. COOPER, E. H. COOPER
School of Electrical Engineering
Purdue University
W. Lafayette, IN 47907

ABSTRACT

A method of designing spread spectrum signal sets with a small value of mean square crosscorrelation is described. Computed results are shown and compared with theoretical bounds.

SUMMARY

In many applications of spread spectrum multiple access systems one is interested in having as large a signal set as possible. In such a situation the mean-square value of the crosscorrelation function for any pair of signals from the set is on the order of 1/TB , where T is the duration of each signal and B is the bandwidth of the signal set. Since the number of simultaneously active users in the sytem is a function of the mean square crosscorrelation, this approach places a limit on system capacity that may be undesirable and unnecessary.

Recent studies have yielded bounds on the number of a signals that can occupy a given time-bandwidth product as function of the mean-square crosscorrelation. These bounds suggest that it may be possible to significantly increase the number of active users in a spread spectrum multiple access system if one is willing to reduce the size of the signal set. There is, in fact, an optimum value of mean square crosscorrelation (for a given value of SNR) at which the number of signals available in the set is equal to the number of active users that can be supported. This condition would be desirable and useful for systems in which the duty factor of each user is very high so that there may be times at which all of the signals in the set may be in use.

The problem that arises in this context is that of designing signal sets for which the mean square crosscorrelation is significantly less than 1/TB. A possible approach to this problem for the case of short frequency hopping sequences employs the concepts of zero- coincidence sequences and spectral shaping to reduce the mean square crosscorrelation. Zero-coincidence sequences are those in which a given frequency is used only once in the entire signal set. Spectral shaping reduces the width of the energy spectrum around each hop frequency. By appropriately combining these techniques it is possible to reduce the mean square crosscorrelation by almost any desired amount. Of course, the number of signals available in a given time-bandwidth product is also reduced.

In order to assess the effectiveness of this approach, the crosscorrelation properties of small sets of signals have been evaluated by computer for all signal pairs within the set. The specific signals considered contained seven frequencies each and employed a raised cosine envelope shape in order to reduce the spectral sidelobes. Signal sets containing 8, 24, and 72 signals are considered. Mean square crosscorrelation values are computed for all pairs of signals and averaged to obtain an effective mean square crosscorrelation. The resulting mean square crosscorrelation values and the corresponding number of signals in the set are then compared with the theoretical bounds. Examples are shown in which the bounds are approached quite closely, while in other cases that appear to be very similar the results are far from the bounds. The conclusion is that although this approach does achieve a reduced mean square crosscorrelation, there are still some unanswered questions in connection with how it should be employed.

ON THE EVALUATION OF THE POWER SPECTRAL DENSITY OF TIME JITTER PRODUCED IN A LINE REGENERATOR

SILVANO PUPOLIN
University of Padova
Department of Electrical Eng.
Padova, Italy

CARLO TOMASI
University of Massachusetts
Dept. of Electrical & Comp. Eng.
Amherst, Massachusetts

Abstract A closed form expression for the spectral density of the time jitter produced in a line regenerator in the presence of a general polynomial nonlinear circuit, bandlimited baseband pulses and an arbitrary tuned filter is carried out. The method applies to PAM digital signal with mutually independent symbol but with an arbitrary number of levels and symbol probabilities. As example of application the spectrum of the time jitter produced in the presence of a fourth power law nonlinear circuit is reported.

INTRODUCTION

Several authors in the last decade have been studying the problem of generation [1-8] of time jitter in synchronous PAM digital data communication systems. Different timing extraction circuits using phase-locked loop [6-7] or simple nonlinear devices and tuned circuits [3-5] have been analyzed. However, they were restricted to considering time limited pulses [1,3,5] or in the case of bandlimited pulses only square law non-linear devices have been taken into account [4]. Since a bandwidth efficient digital transmission is desirable, the trend is to consider digital data systems with bandwidth close to the Nyquist frequency. The time jitter severely degrades the performance of such a system and the square law device in the timing path turns out not to be satisfactory in the extraction of the timing information. Also, the methods proposed to evaluate the time jitter by considering time-limited pulses are too time consuming if we want accurate results.

Here the methos proposed by Franks and Bubrowski [4] for bandlimited pulses is extended to an arbitrary polynomial representation of the non linear circuit.

MODEL OF THE TIMING CIRCUIT

The model of the timing extraction circuit is shown in fig. 1, where s(t) represents a bandlimited PAM signal, $g(\cdot)$ is the impulse response of a prefilter which must be appropriately designed,

$$x(t) = \sum_{n=-\infty}^{+\infty} a_n q(t-nT) \qquad (1)$$

is a PAM signal driving the tuned circuit, a_n is a sequence of mutually independent symbols and T is the signalling period. Finally, we get

---

This work has been supported partially by Consiglio Nazionale delle Ricerche, Italy, and partially by the Ministry of Education, Italy.

$$y(t) = \sum_{k=1}^{N} \alpha_k x^k(t) \quad , \qquad (2)$$

where α_k are the coefficients of the polynomial non-linearity of the zero-memory non-linear device.
$H(f)$ is a bandpass filter tuned to the symbol rate $F \overset{\Delta}{=} 1/T$.

We remark that both $x(t)$, $y(t)$ and $z(t)$, are cyclostationary processes of period T [9,10]. ZCD is a zero-downcrossings detector which gives a sequence of quasi-equally spaced epoches t_n in correspondence of the negative slope zero crossings of $z(t)$. Epoches t_n can be written as follows

$$t_n = t_o + nT + \xi_n T \qquad (3)$$

where t_o is a fixed lag and $\xi_n T$ represents the time jitter.

We remark that ξ_n is a discrete-time random process. In the following we prove that, if $H(f)$ is bandlimited to $(F/2,\ 3/2\ F)$, ξ_n turns out to be a stationary random process and we can compute its autocorrelation

$$R_\xi(kT) \overset{\Delta}{=} E[\xi_n\, \xi_{n+k}] \qquad (4)$$

and its spectral density

$$W_\xi(f) \overset{\Delta}{=} \sum_{k=-\infty}^{+\infty} T\, R_\xi(kT) \exp(-j2\pi fkT) \quad . \qquad (5)$$

The timing wave $z(t)$ can be written as follows:

$$z(t) = [V_o + C_I(t)] \cos[(2\pi F(t-t_1)] + C_Q(t) \operatorname{sen}[(2\pi F(t-t_1)] \qquad (6)$$

where C_I and C_Q are zero-mean jointly wide-sense stationary random processes bandlimited to $F/2$, t_1 and V_o are real constants.

A natural choice of the reference time t_o is represented by the down crossing of the mean of the timing wave $m_z(t) \overset{\Delta}{=} E[z(t)]$, which averaging (6) is given by

$$m_z(t) = V_o \cos[2\pi F(t-t_1)] \qquad (7)$$

so that $t_o = t_1 + T/4$. In such a situation we get:

$$z(t_o + nT) = C_Q(t_o + nT) \quad . \qquad (8)$$

The evaluation of the epoches of zero down-crossings of $z(t)$ is not tractable exactly, however, in practical timing circuits the jitter should be very small and the following approximation has been usually made [1,3,4]:

$$\xi_n = - \frac{z(t_o + nT)}{T\, \dot{m}_z(t_o)} \quad , \qquad (9)$$

where $\dot{m}_z(t)$ is the mean value of the derivative of $z(t)$, namely

$$\dot{m}_z(t) = -\,2\pi F\, V_o \sin\left[2\pi F(t-t_1)\right] \quad . \tag{10}$$

Then, by using (8) and (10) in (9), we get:

$$\xi_n = \frac{C_Q(t_o+nT)}{2\pi\, V_o} \tag{11}$$

By inspection of (11) we remark that the time jitter ξ_n, apart of the scale factor $1/2\pi\ V_o$, is the sampled version of the stationary process $C_Q(t)$. Then its spectral density is given by the periodic repetition of that of C_Q, namely:

$$W_\xi(f) = \frac{1}{(2\pi\ V_o)^2} \sum_{k=-\infty}^{\infty} W_{C_Q}(f-kF) \quad . \tag{12}$$

Since $C_Q(t)$ is a bandlimited to$(-F/2,F/2)$ lowpass random process, the spectral density of the time jitter is given by:

$$W_\xi(f) = \frac{1}{(2\pi\ V_o)^2} W_{C_Q}(f) \quad , \quad |f| < \frac{F}{2} \quad . \tag{13}$$

STATISTICS OF THE TIMING WAVE

Being y(t) a cyclostationary process of period T we define its mean and autocorrelation

$$\begin{aligned} m_y(t) &\overset{\Delta}{=} E\left[y(t)\right] \\ R_y(\tau,t) &\overset{\Delta}{=} E\left[y(t+\tau)y^*(t)\right] \quad , \end{aligned} \tag{14}$$

that are periodic of period T in the argument t, and the spectral density

$$W_y(f,nF) \overset{\Delta}{=} \frac{1}{T}\int_0^T dt \int_{-\infty}^{+\infty} d\tau\ R_y(\tau,t)e^{-j2\pi(f\tau+nFt)} \quad . \tag{15}$$

The spectral density of the timing wave at the output of the filter H(·) is given by:

$$\begin{aligned} W_z(f,nF) &= W_y(f,nF)H(f)H^*(f-nF) \\ M_z(nF) &= M_y(nF)H(nF) \quad , \end{aligned} \tag{16}$$

where

$$M_y(nF) \overset{\Delta}{=} \frac{1}{T}\int_0^T m_y(t)\exp(-j2\pi nFt)dt \quad . \tag{17}$$

Because the filter H(f) is bandlimited only the terms in which the argument of H lies in a neighbourhood of ±F are nonvanishing.

Then the timing wave z(t) turns out to be a sine wave with amplitude and phase modulation. The time jitter is due to the phase modulation, if we do not take into account threshold displacements.

In order to start with the computation we derive first in terms of y(·) and h(·) the parameters characterizing $m_z(t)$ given by eq. (7), i.e.:

$$\begin{aligned} V_o &= 2\left[H(F)\ M_y(F)\right] \quad , \\ t_1 &= -\frac{T}{2\pi}\arg\left[H(F)\ M_y(F)\right] \quad . \end{aligned} \tag{18}$$

IN PHASE AND QUADRATURE COMPONENTS

Eq. (6) giving the expression of the timing wave z(t) can be written in the complex form:

$$z(t) = z_1(t)\exp\left[j2\pi F(t-t_1)\right] + z_{-1}(t)\exp\left[-j2\pi F(t-t_1)\right] \quad , \tag{19}$$

where z_1 and z_{-1} are jointly stationary complex processes related by:

$$z_1(t) = z^*_{-1}(t) = \frac{1}{2}\left[C_I(t) - jC_Q(t) + V_o\right] \quad , \tag{20}$$

with C_I and C_Q jointly stationary real processes.

By recalling the following properties of the spectral density of real processes:

$$\begin{aligned} W_{C_Q}(f) &= W_{C_Q}(-f) \\ W_{C_IC_Q}(f) &= W^*_{C_IC_Q}(-f) \end{aligned} \tag{21}$$

and by using definition (15) we get:

$$\begin{aligned} W_{C_Q}(f) = \operatorname{Re}\{&W_z(f+F,0) + W_z(-f+F,0) - \left[W_z(f+F,2F) + \right. \\ &\left. + W_z(-f+F,2F)\right]\exp(j4\pi F t_1)\} \end{aligned} \tag{22}$$

where t_1 is given by (18).

Hence, the spectral density of the time jitter is obtained by using (22) and (18) in (13).

The value of the spectral density of the time jitter at the origin is very important because a nonzero value means that the time jitter accumulates along a chain of regenerators. As a matter of fact use of eqs. (16), (18) and (22) in (13) yields

$$W_\xi(0) = 2\,\frac{\operatorname{Re}\{W_y(F,0) - W_y(F,2F)\exp\left[-j\arg M_y(F)\right]\}}{|4\pi\ M_y(F)|^2} \tag{23}$$

which clearly shows that $W_\xi(0)$ is independent of the bandpass filter h(t).

Hence, the only action we can made to get $W_\xi(0) = 0$ is to act on the prefilter G(f) or on the nonlinear circuit.

EXAMPLE OF APPLICATION

The method here outlined has been applied to the computation of the spectral density of the jitter produced in a line regenerator by considering squaring and fourth power law nonlinear circuits without prefiltering. We consider as baseband pulse q(t) the one proposed by Franks [11] in order to minimize the time jitter for various values of the roll-off factor β (Fig. 2) and the tuned filter is an LC tank resonant circuit with a mistuning $\delta \overset{\Delta}{=} (f_0T-1)$, where f_0 is the resonant frequency, and a quality factor Q.

The most important result obtained is the evaluation of $W_\xi(f)$ for the fourth power law nonlinear circuit. Inspection of curves reported in figs. 3 also denotes a peculiarity of the fourth power law nonlinearity, $W_\xi(0)$ is positive for $\beta > 0$. This result is particularly important because it is expected it holds also for higher order nonlinearities. The extremely complex expression of $W_z(f,nF)$, even in the simplest case of fourth power law nonlinearity and equilikely input symbols, prevent us to go further in handling the complete formula which gives $W_\xi(f)$. However, the prefilter seems to be required for higher order nonlinearities in order to avoid jitter accumulation. Indeed, as a rule of thumb, in order to get $W_\xi(0) = 0$ seems to be necessary that the bandwidth of q(t) be limited to 2F/N, where N is the order of the nonlinearity and F is the symbol rate.

Finally, more detailed analysis are required in order to optimize the pulse shape q(t). In such an analysis also the effect of threshold displacement must be taken into account.

REFERENCES

[1] W.R. Bennett: "Statistics of regenerative digital transmission", BSTJ, vol. 37, p. 1501-1542, Nov. 1958.

[2] R.D. Gitlin and J. Salz: "Timing recovery in PAM systems", BSTJ, vol. 50, p. 1645-1669, May-June 1971.

[3] L. Bellato and G.L. Cariolaro: "Time jitter in line regenerators with pattern dependent pulse waveforms", Alta Frequenza, vol. 41, p. 800--815, Nov. 1972.

[4] L.E. Franks and J.P. Bubrowski: "Statistical properties of timing jitter in a PAM recovery scheme", IEEE Trans. Commun., vol. COM-22, p. 917--920, July 1974.

[5] G.L. Cariolaro and F. Todero: "A general spectral analysis of time jitter produced in a regenerative repeater", IEEE Trans. Commun., vol. COM-25, p. 417-426, Apr. 1977.

[6] E. Roza: "Analysis of phase-locked timing extraction circuits for pulse code transmission", IEEE Trans. Commun., vol. COM-22, p. 1236-1249, Sept. 1974.

[7] D.L. Duttweiler: "The jitter performance of phase-locked loops extracting timing from baseband data waveforms", BSTJ, vol. 55, p. 37-58, Jan. 1976.

[8] S.U.H. Qureshi: "Timing recovery for equalized partial response systems", IEEE Trans. Commun., vol. COM-24, p. 1326-1330, Dec. 1976.

[9] G. Bilardi and S. Pupolin: "Spectral analysis of the output of a discrete-time input continuous-time output Volterra system driven by white noise", submitted to IEEE Trans. Commun..

[10] W.A. Gardner and L.E. Franks: "Characterization of cyclostationary random signal processes", IEEE Trans. Inform. Theory, vol. IT-21, p. 4-14, Jan. 1975.

[11] L.E. Franks: "Further results on Nyquist's problem in pulse transmission", IEEE Trans. Commun. Technol., vol. COM-16, p. 337-340, Apr. 1968.

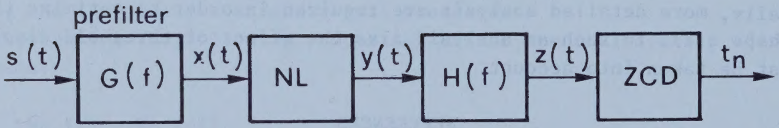


Fig. 1. Model of the timing extraction circuit.

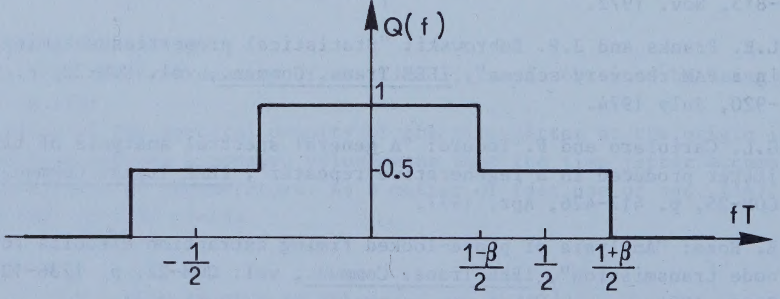


Fig. 2. Fourier transform of the baseband pulse q(t).

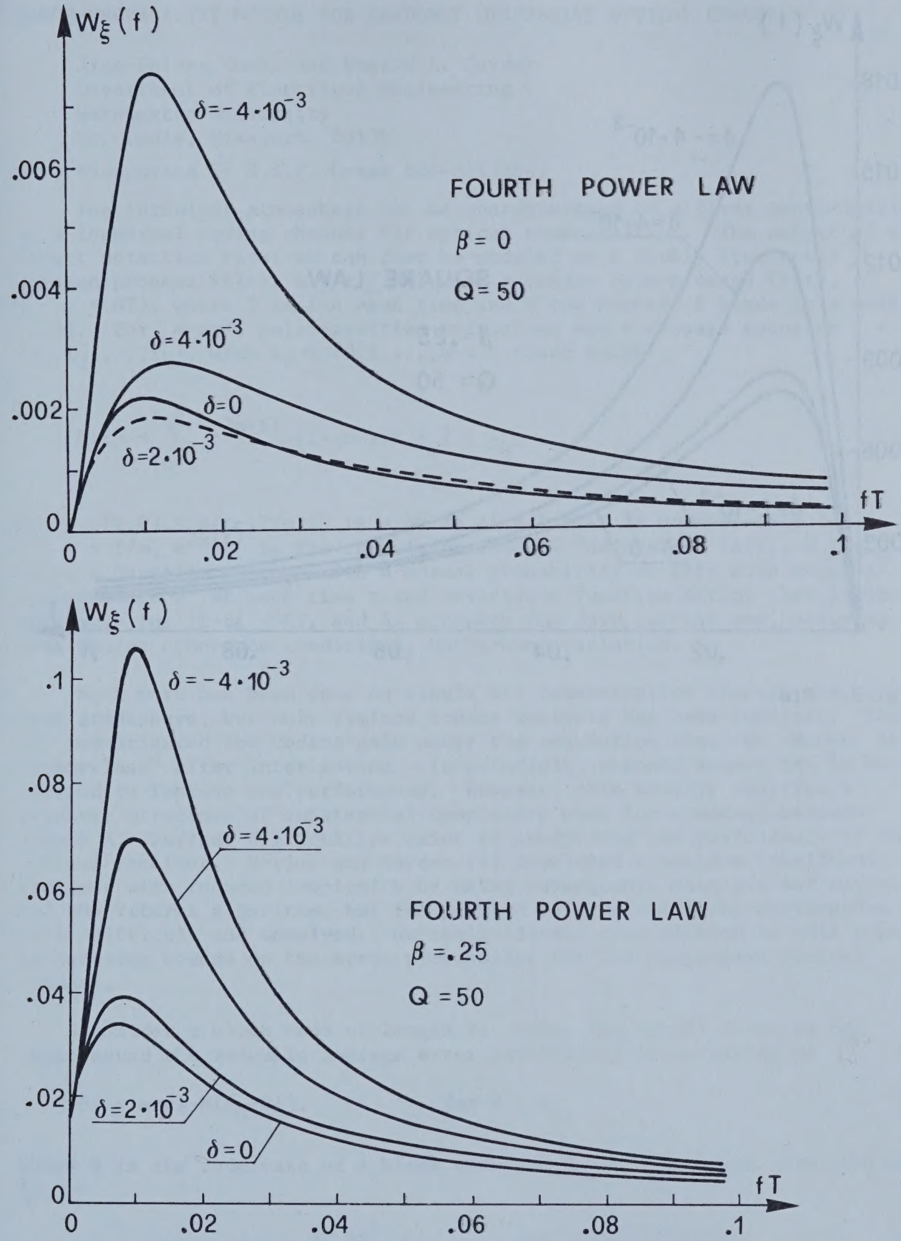


Fig. 3. Spectral density of the time jitter with an LC tuned circuit having a quality factor Q and mistuning δ, for square and fourth power law nonlinear circuit. Baseband pulse: Nyquist with roll-off β.

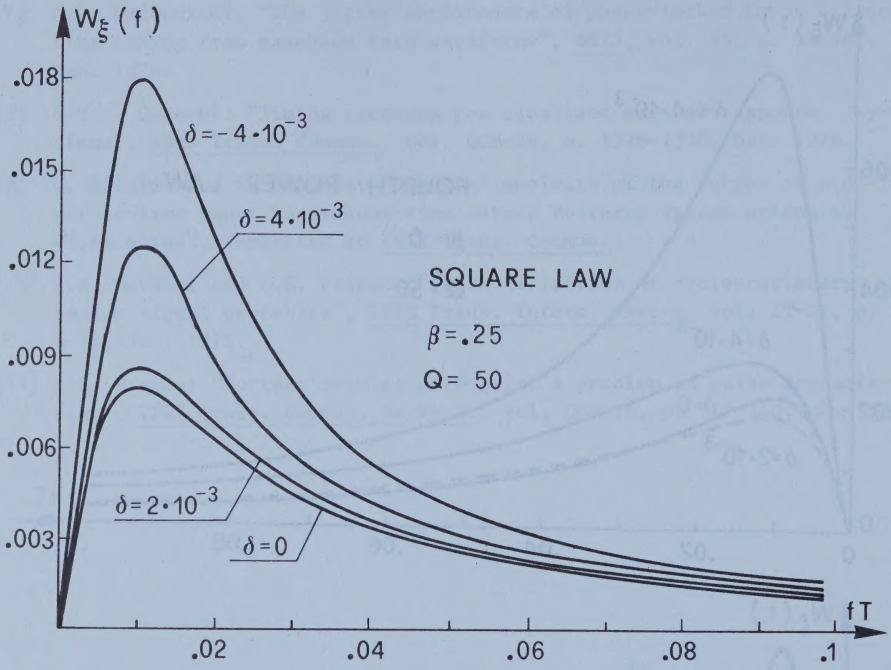

$W_\xi(f)$
.018
.015
.012
.009
.006
.003
$\delta=-4\cdot10^{-3}$
$\delta=4\cdot10^{-3}$
SQUARE LAW
$\beta=.25$
Q= 50
$\delta=2\cdot10^{-3}$
$\delta=0$
fT
.02
.04
.06
.08
.1

Fig. 3. Ctd.

ERROR-PROBABILITY BOUNDS FOR RANDOMLY DISPERSIVE OPTICAL CHANNELS*

Jing-Shiang Cheng and Donald L. Snyder
Department of Electrical Engineering
Washington University
St. Louis, Missouri 63130

*Supported by N.S.F. Grant ECS-8113266.

The turbulent atmosphere can be characterized to a first approximation as a lognormal fading channel for optical communication. The output of a direct detection receiver can then be modeled as a doubly stochastic Poisson process $\{N(t),\ 0 \leq t \leq NT\}$ with a random rate-process $\{\lambda(t),\ 0 \leq t \leq NT\}$, where T is the baud time and N the number of bauds in a message [1,2]. For m-ary, pulse-position modulation and a message sequence $I = [i_1, i_2, \ldots, i_N]$, with $i_N \in \{0,1,\ldots,m-1\}$, there holds

$$\lambda(t) = \sum_{n=1}^{N} e^{2x(t)} p[t-(n-1)T, i_n] + \lambda_0,$$

where $p[t,0] = p[t-iT/m,i]$ is a pulse signal that is nonzero only for $0 \leq t < T/m$, $e^{2x(t)}$ is the turbulence induced fading with $\{x(t),\ 0 \leq t \leq NT\}$ being a Gaussian process with a normal probability density with mean $-\sigma^2$ and variance σ^2 at each time t and covariance function K(t,u) that is nonzero only for $|t-u| \leq LT$, and λ_0 accounts for dark current and, assuming weak photon coherence conditions, background radiation.

Much work has been done on single bit communication through the turbulent atmosphere, but only limited coding analysis has been reported. Chan [3] investigated the coding gain under the assumption that the channel is "memoryless" after interleaving. In principle, channel memory can be exploited to improve the performance. However, this usually requires a receiver structure of substantial complexity even for a modest message length N. Further difficulties exist in predicting the performance of the optimum receiver. Morley and Snyder [4] developed a maximum likelihood receiver with reduced complexity by using subsequence waveform estimators and the Viterbi algorithm, but the problem of predicting the performance is still difficult and unsolved. We shall address this problem in this paper by deriving bounds on the error probability for the dispersive optical-channel.

Consider a block code of length N. Using the cutoff rate, we can upper bound the ensemble average error probability Pe according to

$$Pe \leq \exp[-N(R_0-R)], \qquad \text{for } R \leq R_0 \tag{1}$$

where R is the code rate of a block code and R_0 is the cutoff rate, defined by

$$R_0 = -(1/N)\ \ln\{\min_q \sum_i \sum_j q(v_i)q(v_j) \sum_y \sqrt{p(y|v_i)p(y|v_j)}\}, \tag{2}$$

where the last sum is the Bhattacharyya bound on the error probability for decoding v_i and v_j, and where $q(v_i)$ is the probability of message v_i.

To make the problem analytically tractable, we assume that x(t) is a piecewise constant process, $x(t) = x_k$ for $kT \leq t < (k+1)T$. The conditional Bhattacharyya bound given $x = [x_1,x_2,\ldots,x_N]$ can then be expressed as

$$\sum_y \sqrt{p(y|v_i)p(y|v_j)} = (1/2)\exp[-(1/2)\{\sqrt{\lambda_s \exp(2x_k)+\lambda_b} - \sqrt{\lambda_b}\}^2], \tag{3}$$

where λ_s is the average signal count per channel symbol and λ_b is the average noise count per slot.

For m-ary PPM, the minimizing distribution q is found to be the uniform distribution $q(v_i) = m^{-N}$. Thus, the cutoff rate can be written as

$$R_0 = \ln(m) - (1/N)\ln\{\int \prod_{k=1}^{N} (m-1)y_k+1]p(\underline{x})d\underline{x} - 1\} \tag{4}$$

where

$$y_k = \exp -(1/2)[\sqrt{\lambda_s \exp(2x_k)+\lambda_b} - \sqrt{\lambda_b}]^2$$

$$\approx \begin{cases} \exp[-(1/2)\lambda_s \exp(2x_k)] & \text{if } \lambda_s/\lambda_b \gg 1 \\ \exp[-(1/8)\lambda_b \lambda_s^2 \exp(4x_k)] & \text{if } \lambda_s/\lambda_b \ll 1 \end{cases}$$

and $p(\underline{x})$ is the multivariate normal density with mean vector $\underline{m}$ covariance matrix $\underline{K}$.

If the fading is very slow, $x(t) = x$ throughout the message interval, and the product in (4) becomes $[1+(m-1)y]^N$. Then, the bound in (4) simplifies to the form obtained by Nakai [6] and is similar to the form given by Shapiro and Harney [7], which can be expressed in terms of the frustration function. An efficient algorithm for evaluating the simplified expression is also given by Shapiro and Harney.

In general, the evaluation of the bound in (4) is very difficult analytically. However, we can use saddle point methods from the literature [8].

1. D.L. Snyder, Random Point Processes, Wiley, 1975.
2. J.H. Shapiro, "Imaging and Optical Communication Through Atmospheric Turbulence," Ch. 6 in: Laser Beam Propagation in the Atmosphere (J.W. Strohbehn, ed.), Springer, 1978.
3. V.W.S. Chan, "Coding for the Turbulent Atmospheric Optical Channel," NTC'79 Conf. Record, pp. 44.3.1-44.3.5, 1979.
4. R.E. Morley and D.L. Snyder, "Maximum Likelihood Sequence Estimators for Randomly Dispersive Channels," IEEE Trans. on Comm., vol. COM-27, pp. 833-839, June 1979.
5. J.W. Wozencraft and J.M. Jacobs, Principles of Communication Engineering, Wiley, 1965.
6. J. Nakai, "Coding and Modulation Analysis for Optical Communication Channels," Ph.D. Thesis, Dept. of Elec. Engineering, M.I.T., 1982.
7. J.H. Shapiro and R.C. Harney, "Simple Algorithms for Calculating Optical Communication Performance Through Turbulence," Proc. SPIE, vol. 295, San Diego, 1981.
8. C.W. Helstrom, "Approximate Evaluation of Detection Probabilities in Radar and Optical Communication," IEEE Trans. on Aero. and Elect. Sys., vol. AES-14, pp. 630-640, 1978.

ON THE APPLICATION OF QUANTISED FEEDBACK AND PARTIAL RESPONSE TECHNIQUES TO HIGH BIT-RATE OPTICAL FIBRE COMMUNICATIONS

John J. O'Reilly, M. Duarte and L. Blank
Department of Electrical Engineering Science,
University of Essex, U.K.

Abstract

The use of quantised feedback and partial response techniques as a means of combatting the strong high frequency noise terms encountered in receivers for high bit rate optical fibre communications are examined. In particular, it is suggested that the integrating effect of the widely adopted high impedance receiver structure can usefully be exploited in this context. This provides a beneficial reduction in equalisation requirements compared with conventional receivers. It is shown that significant improvement can be obtained and the practicability of the proposal is substantiated by reference to an experimental system.

1. Introduction

High sensitivity receiver front-ends adopted for optical fibre communications exhibit strong high frequency noise characteristics which tend to be the limiting noise source for high bit-rate systems. This is especially true for second generation systems operating in the second and third windows of silica fibres, at wavelengths of 1300nm and 1550nm, since PIN photodiode receivers are most commonly adopted here due to the poor gain noise and leakage characteristics of currently available avalanche photodiodes for these wavelengths.

In view of this and the continuing trend towards yet higher signalling rates, it is appropriate that we examine carefully the prospects for adopting receiver structures specifically suited to combat high frequency noise. For a conventional receiver structure based on a front-end followed by an equalising filter/pulse shaping network and a decision/retiming device some degree of optimisation of the filter is possible. This generally takes the form of a trade-off between noise and intersymbol interference (isi). More heavily filtering the noise admitts isi and the filter is optimised for minimum noise+isi variance[1]. An alternative approach operates directly with a bound on error probability[2] with broadly similar results for PIN receivers.

In this contribution we consider instead the use of alternative receiver forms which offer the prospect of more effectively combatting high frequency noise. The two techniques, quantised feedback and partial response, are considered separately and in combination. We find that each individually has merits and that their combined use can offer a significant performance advantage. Finally, we note that the move to yet higher bit-rates may lead to the dominance of $(\text{frequency})^4$ noise components in which case these alternative structures offer further advantages.

2. The Optical Receiver Front-end

We consider a high-impedance or 'integrating' front-end of the form shown in figure 1. The photodiode spreading resistance R_s and its associated noise spectral density, S_s, are usually neglected for low and medium bit-rates (in a typical photodiode $R_s \sim 10\Omega$). However for bit-rates of several hundred Mbits/sec its relative importance starts to be noticeable, and as such it will be included in our analysis.

The arguments in favour of this receiver structure are available in the literature[3,4], and we note here simply that the resistance R_b is chosen to be as large as feasible in order to minimise its thermal noise contribution and to maintain as high as possible the total impedance presented to the photodiode. This results in the detected signal being approximately integrated by the first-order low-pass section, with a very low cut-off frequency, formed by R_b and the joint effects of C_{sa} and C_j (the spreading resistance effects on the signal shape can be neglected even when its noise contribution starts being noticeable). In a conventional design[1] the equaliser corrects for this by performing a differentiation over the appropriate frequency range and as a consequence imparts $(\text{frequency})^2$ weighting to the output noise terms arrising from the input noise sources (S_s can be translated into an equivalent voltage source with $(\text{frequency})^2$ dependence in series with S_e). As a result the noise power density after equalisation is made up of white plus $(\text{frequency})^2$ plus $(\text{frequency})^4$ terms associated respectively with S_i, S_e and S_s. We should note at this stage that this noise characteristic is not unique to the high-impedance plus equaliser front-end: an alternative approach makes use of a transimpedance configuration[4], but once again the combination of the capacitances C_j and C_{sa} with the diode plus amplifier equivalent voltage noise sources gives rise to a similar noise power distribution at point (b) of figures 1 and 2.a.

3. The Quantised Feedback Approach

Referring once more to figure 1, a study by Goell[5] provides a detailed description of the noise components in the high-impedance front-end. The expressions derived by this author are complicated but once referred to point (a) of figure 1 can be reduced to the form:

$$N(f) = S_i A^2 R_1{}^2/[1+(2\pi R_1 C_1)^2 f^2]+S_e\ A^2+S_s A^2 R_2{}^2[(2\pi R_2 C_2)^2 f^2] \qquad \ldots(1)$$

where C_1, C_2, R_1 and R_2 are in some way related to C_j, C_{sa}, R_s and R_b. For convenience we employ a compact version of (1):

$$N(f) = (\mu/2)/[1+(f/f_{c1})^2]+(\eta/2)+(\sigma/2)[f/f_{c2}]^2 \qquad \ldots(2)$$

where μ, η and σ, f_{c1} and f_{c2} can be found by equating the low-frequency, white and $(\text{frequency})^2$ terms of (1) and (2).

As previously mentioned, we note that at this same point the 'integrated' signal suffers extreme isi corruption. This can, however, be corrected very simply by using a quantised (decision directed) feedback(QF) arrangement as outlined in figure 2.b. Some forward path bandlimiting prior to detection is required too, but the requirements here are relatively modest since the filter is now required to combat basically $(\text{frequency})^2$ and white noise, instead of $(\text{frequency})^4$ and $(\text{frequency})^2$ as would be the case for a conventional receiver, such as that of figure 2.a. Figure 2.c illustrates the signal element formats in the forward and feedback paths, from which the basic philosophy of the QF arrangement should be apparent.

This simple, non-optimal scheme has a number of attractive features which we note before discussing improvements. Firstly, the majority of the forward filtering is provided by the integrating front-end itself. This allows the subsequent noise limiting filters to be of rather low complexity. Secondly, the feedback path filtering is similarly simple, being predominantly a CR integrator. Finally, the loop is self-stabilising so that error extension due to this integrating QF scheme is minimal and well-defined. For error rates of practical interest the error extension is precisely two: a single noise-induced decision error gives rise to just one further error due to the feedback loop. This behaviour is readily predicted by considering possible signal and error sequences and has been confirmed experimentally[6]. Signal waveforms relating to this simple scheme, obtained from the same experimental system, are shown in figures 3.a and b. Figure 3.c depicts one possible error situation and illustrates the recovery mechanism.

While this system is not optimum, it has been found to offer an improvment of upto 3dB in signal to noise ratio (μ noise assumed dominant) compared with a conventional system providing the same eye pattern. Conventional optical receivers generally employ 100% raised cosine (RC) target element spectrum, however, so a camparison of QF and conventional systems has been effected on these grounds too. The results were obtained by numerical evaluation and are shown in table 1 for various bit rates under the 'full-response' designation. To facilitate the interpretation of these results we detail below the underlying assumptions.

(i)It was assumed that the input signal element $h_i(t)$ in both the conventional and QF receiver was a constant amplitude non-return to zero pulse of the form:

$$h_i(t) = \text{rect}(t/T) = \begin{cases} 1, & |t|<T/2 \\ 0 & \text{elsewhere} \end{cases}$$

where T = symbol duration = $(\text{bit-rate})^{-1}$.

This situation of a dispersion-free pulse with amplitude kept constant with bit-rate is appropriate to recent monomode and certain graded index optical fibre systems where dispersion is nearly negligible and where the transmitter source is peak power limited. On account of this last characteristic we must bear in mind when interpreting table 1 that as the bit rate increases the energy per received optical pulse decreases in the same proportion, a situation somewhat different from that usually assumed when studying the performance of common electrical digital transmission systems.

(ii)The bandlimiting and shaping filters $H_c(f)$ and $H_q(f)$ were selected to produce at the input of the decision circuit signal elements with 100% raised cosine spectra and with equal amplitudes at the decision times.

(iii)For the cut-off frequencies f_{c1} and f_{c2} we assumed values typically encountered in modern receivers:

$$f_{c1} = 30\text{kHz}$$

$$f_{c2} = 100\text{MHz}$$

(iv)The units of μ, η and σ are $(\text{volts})^2/\text{MHz}$.

Within the limits of these assumptions we conclude from table 1 that QF offers a significant advantage over conventional designs if either or both of the η or σ noise dominate, as is generally the case for high bit-rate systems.

4. Quantised Feedback plus Partial Response

An alternative means of combatting $(\text{frequency})^2$ and $(\text{frequency})^4$ noise at the equaliser output of the conventional receiver is to employ a filter giving rise to a class I partial response characteristic. The benefit obtained is greater if there is significant dispersion in the optical channel[7] but for high bit-rate systems utilising PIN receivers there is an advantage even if dispersion is minimal, as is the case of recent monomode systems[8 and table 1 of this paper: conventional receiver, partial response]. This sugests that the combined use of QF and PR may be appropriate . That this is indeed the case can be seen from the illustrative results of Table 1. The remarks and assumptions made in the previous section concerning the received optical pulse shape, and the role of the bandlimiting and shaping filters in order to ensure constant decision distances, apply here also.

There is a further, less apparent, advantage of PR in this context. At high signalling rates QF loop delay can degrade performance: the PR system is more tolerant to loop delay than its full-response counterpart.

5. Concluding Remarks

We have shown that Quantised Feedback and Partial Response, used separately and in combination, can provide significant improvement compared with conventional receivers for high bit-rate optical communications. With typical present day devices signal to noise ratio improvements approaching 4.2 dB and 13 dB in, respectively, η and σ noise dominated systems can be obtained using the combination of QF and PR techniques instead of a conventional design. In any particular application these conclusions will be sensitive to the precise balance of the various noise sources and to the receiver filtering scheme.

6. Acknowledgments

One of the authors, M. Duarte, is grateful to the Calouste Gulbenkian Foundation, the Comissão Permanente INVOTAN and the University of Aveiro(Portugal) for financial support in relation to the work reported here.

7. References

1.O'Mahony, M.J.:'Signal design for optical fibre communication', Digest of Colloquium: Mathematical Topics in Telecommunications - Calculus of Variations, University of Essex, Nov. 1980, pp39-46.

2.Rocha, J.R.F. and O'Reilly, J.J.:'The Chernoff bound as a basis for optical fibre receiver design', Digest of Annual Conference on Information Science and Systems, Princeton, March 1982.

3.Personick, S.D.:'Receiver design for digital fibre optic communication systems', B.S.T.J. vol.52, No.6, (July-August, 1973), pp843-886.

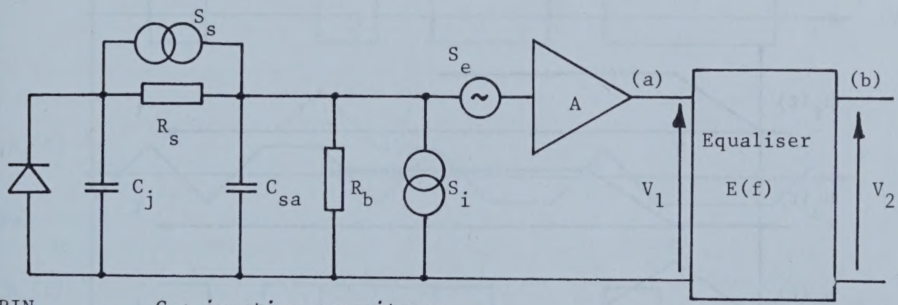
4.House, K.:'Detection and Equalisation in Fiber-Optic Communication', Ph.D. Thesis, University of Essex, December 1979.

5.Goell, J.E.:'Input amplifiers for optical PCM receivers', B.S.T.J. vol53, No. 9, November 1974, pp1771-1793.

6.Blank, L.:'A Decision Directed Feedback Receiver for Digital Optical Communication Systems', Internal Report, University of Essex, July 1982.

7.Sporndli, J.R.:'The influence of intersymbol interference on the performance of optical receivers', M.Sc. dissertation, University of Essex, May 1977.

8.O'Mahony, M.J.:'Duobinary transmission with pin-fet optical receivers', Electron. Lett. vol16, 1980, p752.



PIN
photodiode
(current
source)

C_j: junction capacitance

C_{sa}: stray and amplifier capacitance capacitance

R_s: photodiode spreading resistance

R_b: bias resistance

Figure 1. Schematic of high-impedance or 'integrating' front-end.

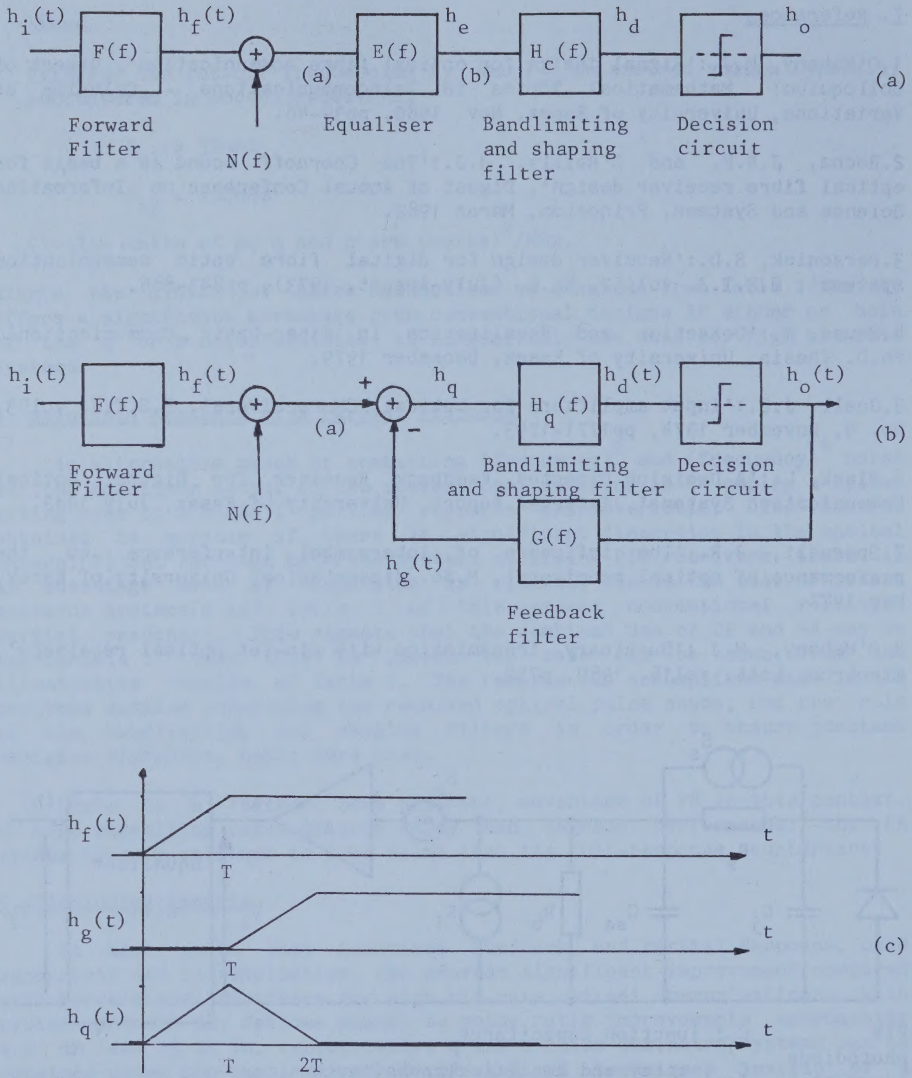
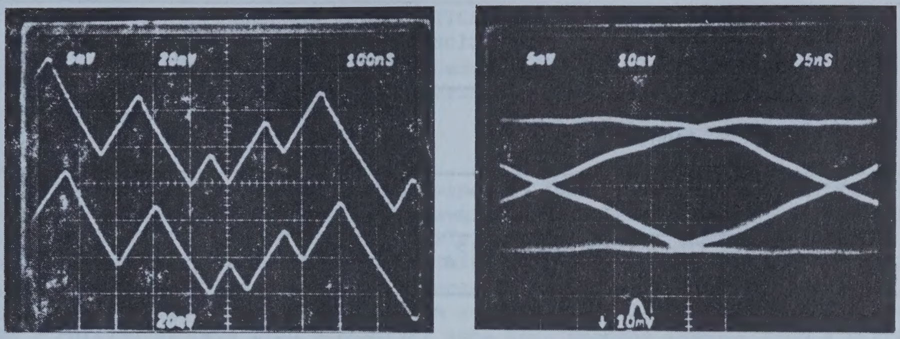

$h_i(t)$
F(f)
$h_f(t)$
+
E(f)
h_e
(a)
(b)
$H_c(f)$
h_d
h_o
(a)
Forward Filter
N(f)
Equaliser
Bandlimiting and shaping filter
Decision circuit
$h_i(t)$
F(f)
$h_f(t)$
+
(a)
+
h_q
+
–
$H_q(f)$
$h_d(t)$
$h_o(t)$
(b)
Forward Filter
N(f)
Bandlimiting and shaping filter
Decision circuit
G(f)
$h_g(t)$
Feedback filter
$h_f(t)$
t
T
$h_g(t)$
t
T
(c)
$h_q(t)$
t
T
2T

Figure 2. (a) Conventional receiver, (b) Quantised feedback receiver, (c) Basic waveforms in a QF receiver with the forward path filter dominated by the integrating front-end.



(a) Signal waveforms.
Upper trace: forward path signal
Lower trace: feedback signal

(b) System eye pattern

(c) Error propagation and
system recovery mechanism
with a single decision error

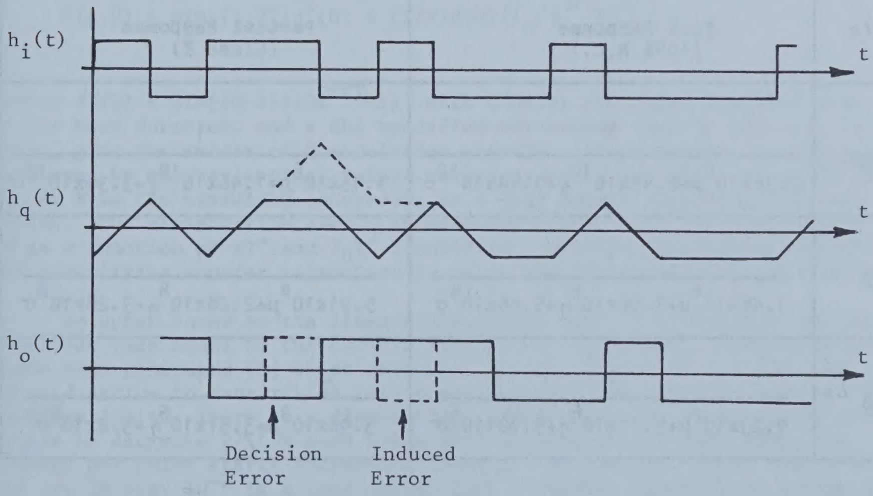

$h_i(t)$
t
$h_q(t)$
t
$h_o(t)$
t
Decision Error
Induced Error

Figure 3. Illustrative signal waveforms for an experimental Quantised Feedback system exploiting the integrating front-end as the primary forward filtering.

Rate Mb/s	Conventional Receiver	
	Full Response (100% R.C.)	Partial Response (Class I)
2240	$\frac{1}{44.8\mu+3.85\text{x}10^{10}\eta+6.76\text{x}10^{12}\sigma}$	$\frac{1}{89.7\mu+1.93\text{x}10^{10}\eta+8.46\text{x}10^{11}\sigma}$
560	$\frac{1}{11.22\mu+6.02\text{x}10^{8}\eta+6.6\text{x}10^{9}\sigma}$	$\frac{1}{22.43\mu+3.01\text{x}10^{8}\eta+8.26\text{x}10^{8}\sigma}$
140	$\frac{1}{2.8\mu+9.40\text{x}10^{6}\eta+6.45\text{x}10^{6}\sigma}$	$\frac{1}{5.6\mu+4.7\text{x}10^{6}\eta+8.06\text{x}10^{5}\sigma}$

Rate Mb/s	Quantised Feedback Receiver	
	Full Response (100% R.C.)	Partial Response (Class I)
2240	$\frac{1}{2.36\text{x}10^{5}\mu+2.43\text{x}10^{10}\eta+5.99\text{x}10^{12}\sigma}$	$\frac{1}{9.45\text{x}10^{6}\mu+1.46\text{x}10^{10}\eta+3.36\text{x}10^{11}\sigma}$
560	$\frac{1}{1.48\text{x}10^{4}\mu+3.66\text{x}10^{8}\eta+5.86\text{x}10^{9}\sigma}$	$\frac{1}{5.91\text{x}10^{4}\mu+2.28\text{x}10^{8}\eta+3.28\text{x}10^{8}\sigma}$
140	$\frac{1}{9.23\text{x}10^{2}\mu+5.71\text{x}10^{6}\eta+5.72\text{x}10^{6}\sigma}$	$\frac{1}{3.96\text{x}10^{3}\mu+3.57\text{x}10^{6}\eta+3.2\text{x}10^{5}\sigma}$

TABLE 1: Signal to noise ratios in conventional and Q.F. receivers with normalized input optical power and decision distances.

SIGNAL DESIGN FOR POWER EFFICIENT OPTICAL COMMUNICATION*

Donald L. Snyder and Costas N. Georghiades
Department of Electrical Engineering
Washington University
St. Louis, Missouri 63130

The problem we address in this paper is that of designing the signal set for modulation in an optical communication system. The signals as well as the number of signals are to be selected to minimize the average power required to communicate over a bandlimited optical channel at a specified throughput rate. The channel introduces a nonzero dark current and a background radiation under low photon coherence conditions; this is modelled as a Poisson process with a constant rate λ_0. Encoder symbols from a Q-ary alphabet are the input to the modulator at a rate of 1/T symbols per second. We assume no output quantization for the demodulator. The combination of the modulator, channel, optical detector, and demodulator form a discrete memoryless channel that has been studied extensively in the literature. Recent papers that are most relevant to the work presented herein are those of Snyder and Rhodes [1], McEliece [2], Massey [3], and Butman, Katz, and Lesh [4]. Compared to [1], we introduce a bandwidth constraint and seek to minimize average power. Compared to [4], we do not assume PPM signalling, and we include nonzero noise.

We show that Q-ary PPM with raised cosine pulses is optimum. Specifically, we show that the average power is lower bounded according to

$$P(\underline{g},Q) \geq \min_{Q}\{(1/2T)d^2(Q) + (2/T)d(Q)[\lambda_0/3^{3/2}B]^{1/2},$$

where $d^2(Q) = 2\ln[(Q-1)/(Qe^{-rT}-1)]$ with Q being the input alphabet size, T the baud duration, and r the specified throughput rate in nats per second. Also, $\underline{g}$ is the vector of Q modulation signals. The quantity in braces is achieved by nonoverlapping, raised cosine pulses of duration $T' = 1/\sqrt{3B}$, where B is the bandwidth constraint in a mean square definition of bandwidth. The minimization over Q is also addressed, with the best values of Q as a function of rT' and $\lambda_0 T'$ identified. Finally, we examine the problem of identifying encoder parameters to match the power efficient modulator.

An upper bound on the dimensionless code rate is obtained by setting the code rate equal to the cut-off rate in nats per baud. Various graphs have been generated [5] which show that for all values of $\lambda_0/\sqrt{3B}$, one should strive to have $r/\sqrt{3B}$ less than about 10^{-2} in order to approach the optimum limit. There is a diminishing return, however, so it is not worthwhile to decrease $r/\sqrt{3B}$ much below 10^{-2} because at the same time, the energy per pulse starts increasing rapidly. It appears, then, that a value of $r/\sqrt{3B}$ near 10^{-2} is a good design goal. Another point worth noting is that an increase in the baud time, T, results in an increase in average power, a result which seems surprising at first. This is due to the constraint that r is fixed. Finally, we note that increasing the bandwidth B results in a two-fold benefit in performance because the performance improves as both $r/\sqrt{3B}$ and $\lambda_0/\sqrt{3B}$ decrease.

---

*This work was supported by the National Science Foundation under Grant ECS-8113266.

References

1. D. L. Snyder and I. B. Rhodes, "Some Implications of the Cutoff-Rate Criterion for Coded Direct-Detection Optical Communication Systems," IEEE Trans. on Information Theory, vol. IT-26, no. 3, pp. 327-338, May 1980.

2. R. J. McEliece, "Practical Codes for Photon Communication," IEEE Trans. on Information Theory, vol. IT-27, no. 4, pp. 393-398, July 1981.

3. J. L. Massey, "Capacity, Cutoff Rate, and Coding for a Direct Detection Optical Channel," IEEE Trans. on Communications, vol. COM-29, no. 11, pp. 1615-1621, November 1981.

4. S. A. Butman, J. Katz, and J. R. Lesh, "Bandwidth Limitations on Noiseless Channel Optical Channel Capacity," IEEE Trans. on Communications, vol. COM-30, no. 5, pp. 1262-1264, May 1982.

5. D. L. Snyder and C. N. Georghiades, "Design of Coding and Modulation for Power-Efficient Use of a Bandlimited Optical Channel," IEEE Trans. on Communications (in review).

ON THE BEHAVIOUR OF THE DATA QUEUE IN A VOICE/DATA INTEGRATED SYSTEM

DAVID D.W. YAO
Department of Industrial Engineering
University of Toronto
Toronto, Canada

A. LEON-GARCIA
Department of Electrical Engineering
University of Toronto
Toronto, Canada

ABSTRACT

The diffusion approximation model of [11] is refined such that it will give good approximate results for all traffic intensities and exact result for single channel systems. The refined model is then used to study both the transient and equilibrium behaviours of the data queue.

1. INTRODUCTION

We consider a voice/data integrated system, where voice (data) messages arrive following a Poisson stream with rate λ_v (λ_d), and require an exponential service with rate μ_v (μ_d). There are v (d) parallel channels serving voice (data) messages. While data messages can utilize any unoccupied voice channels, voice messages are not allowed to use the dedicated data channels. Voice messages are not allowed to buffer; however, upon arrival they can preempt the service of any data message occupying a voice channel. Let $\rho_v \equiv \lambda_v/\mu_v$ and $\rho_d \equiv \lambda_d/\mu_d$ be the traffic intensities of the voice and the data messages, respectively.

Obviously, the service of voice messages in the system described above can be modelled as an M/M/v loss system, which operates independently of the data messages, with steady-state probabilities known as follows:

$$\pi_0 = \left[\sum_{k=0}^{v} \frac{\rho_v^k}{k!} \right]^{-1} \qquad \pi_i = \frac{\rho_v^i}{i!} \pi_0 (i = 1,2,\ldots v) \tag{1.1}$$

However, the congestion of the data queue depends heavily on the arrival and service parameters of the voice messages. So the concentration of this paper will be on the study of the behaviour of the data queue. Before we proceed further, it should be noted that throughout the paper the term "data queue" denotes the total data messages in the system, i.e. including those in service.

The difficulties involved in the study of this type of system have long been recognized. The magnitude of the ratio μ_d/μ_v (typically having the order of 10^4) can even make a simulation study extremely difficult ([8]). Recently, it is known that the equilibrium behaviour of the above system can be computationally analysed using the matrix-geometric approach ([12, 14]), which, however, becomes less efficient when the number of channels increases.

Another limitation of this approach is that it does not seem applicable to the study of the transient behaviour of the data queue, which, in this particular type of system, is often more crucial than the equilibrium phenomena ([8]). Actually, for analysing most real systems, approximations seem to be the only practical and efficient means. Available in current literature are the fluid approximation ([8]) and the diffusion approximation ([11]).

The significant contribution of [11] is that (using the analytic approach developed in [2]) the authors have shown that the marginal data process can be approximated by a one dimensional Wiener process with drift $-\mu_d(1-\rho)(v+d)$ and variance

$$2\mu_d\left[\rho_d + \frac{\mu_d}{\mu_v\rho_v}\sum_{i=0}^{v-1}\frac{T_i^2}{\pi_i}\right] \tag{1.2}$$

where

$$\rho \equiv [\rho_d + \rho_v(1-\pi_v)]/(v+d), \qquad T_i \equiv \sum_{k=0}^{i}\pi_k(k-\rho_v\,(1-\pi_v)) \tag{1.3}$$

The Wiener process is also characterized with a reflecting barrier at zero and thus follows an exponential distribution. With this approach the authors proceed to suggest an approximate formula for the equilibrium data queue length as follows:

$$\hat{D} = \frac{\rho_d + \dfrac{\mu_d}{\mu_v\rho_v}\displaystyle\sum_{i=0}^{v-1}\frac{T_i^2}{\pi_i}}{(1-\rho)\ (v+d)} \tag{1.4}$$

The interest of [11] is , however, mainly analytical. Numerically, the proposed approximation only offers good accuracy for traffic intensities higher than .95 (as indicated by our numerical studies as well as by the same authors in [12]). It is the aim of the present paper to refine the model of [11] so that it will give good approximate results for all traffic intensities. The refined model will then be used to study the equilibrium as well as transient behaviours of the data queue. Further extensions of the model will also be discussed briefly.

2. THE MODEL AND EQUILIBRIUM BEHAVIOUR

We consider the equilibrium data queue process $\{D\}$. Our point of departure is the main conclusion of [11], i.e., the marginal data process can be approximated by a Wiener process with certain drift and variance. We will restrict our approximation model to systems without dedicated data channels, i.e., d=0. For real systems with d>0, we suggest the study of an "equivalent" system with d=0 while the data traffic is adjusted to ρ_d-d (ρ_d is preassumed to be greater than d, since the opposite case is usually not of interest to this type of system). So we have $\beta \equiv -v\mu_d(1-\rho)$ as the drift of the Wiener process.

Next we consider the variance of the Wiener process. Since this is the sum of the variances of the input and output processes of the system, and since data messages arrive following a Poisson process, it can be expressed as follows:

$$\alpha \equiv 2(\lambda_d + E\mu_d KS^2) \qquad (2.1)$$

where $E=v-\rho_v(1-\pi_v)$ is the effective (average) number of channels serving data messages and KS^2 is the squared coefficient of variation (SCV) of this (superposition) service time process. The second term on the RHS of (2.1) is formulated by assuming that the superposition service process is renewal and then using asymptotic results of renewal processes (see e.g. [3]). Comparing (2.1) with (1.2), we have

$$KS^2 = E^{-1} \cdot \frac{\mu_d}{\mu_v \rho_v} \sum_{i=0}^{v-1} \frac{T_i^2}{\pi_i} \qquad (2.2)$$

However, it is natural to expect better results if we model the output process by the departure process instead of the service process. The SCV of the former can be expressed as follows ([5, 15])

$$KD^2 = (1-\theta^2) + \theta^2 KS^2 \qquad (2.3)$$

where $\theta=\rho_d/E$ is the data messages' utilization of the channels. So replacing KS^2 by KD^2 in (2.1), we have the variance of the Wiener process as follows:

$$\alpha = 2\mu_d \left[\rho_d + E(1-\theta^2) + \theta^2 \cdot \frac{\mu_d}{\mu_v \rho_v} \sum_{i=0}^{v-1} \frac{T_i^2}{\pi_i} \right] \qquad (2.4)$$

We prefer to write α in the following form:

$$\alpha = 2\mu_d \left[\rho_d + ((1-\theta^2)/v^a + \theta^2) \cdot \frac{\mu_d}{\mu_v \rho_v} \sum_{i=0}^{v-1} \frac{T_i^2}{\pi_i} \right] \qquad (2.5)$$

where a satisfies $v^a = E^{-1} \cdot \frac{\mu_d}{\mu_v \rho_v} \sum_{i=0}^{v-1} \frac{T_i^2}{\pi_i}$.

Now the remaining task is to determine the appropriate diffusion equation and boundary condition. We use the more general theory of Feller instead of that of Kolmogorov to study the Wiener process([1]): the boundary zero is treated as a "visiting barrier" with an elementary return process ([6]), i.e., whenever the Wiener process hits the boundary it will remain there for an exponentially distributed random time instead of reflecting back instataneously (as is the case in traditional "reflecting barrier" approaches). Obviously, this approach models more precisely the behaviour of the data queue at the boundary. With this approach, the pdf of the Wiener process satisfies a system of differential equations (p. 146 of [1]), the equilibrium version of which is as follows (see also [9, 10]):

$$\frac{1}{2}\alpha f''(x) - \beta f'(x) = -\lambda_d p_0 \delta(x-1) \qquad (2.6)$$

$$\lim_{x \to 0} [\frac{1}{2}\alpha f'(x) - \beta f(x)] = \lambda_d p_0 \qquad (2.7)$$

$$\lim_{x\to 0} f(x) = 0 \tag{2.8}$$

where $\delta(x-1)$ is the Dirac's delta function, and $p_i (i=0,1,\ldots)$ is the steady-state probability densities of the data queue. Let $\gamma=2\beta/\alpha$. It is straightforward to derive the solution $f(x)$ as follows:

$$f(x) = \begin{cases} p_0 \dfrac{\lambda_d}{\beta}(e^{\gamma x} - 1) & 0 \le x \le 1 \\[2ex] p_0 \dfrac{\lambda_d}{\beta}(1 - e^{-\gamma})e^{\gamma x} & x > 1 \end{cases} \tag{2.9}$$

From the normalization condition and the discretization of $f(x)$:

$$p_0 + \int_0^\infty f(x)dx = 1, \qquad p_i = \int_{i-1}^{i} f(x)dx \tag{2.10}$$

we have immediately the probabilities as follows:

$$p_0 = \frac{\beta}{\beta - \lambda_d}, \qquad p_1 = p_0 \frac{\lambda_d}{\beta}\left(\frac{e^{\gamma} - 1}{\gamma} - 1\right) \tag{2.11}$$

$$p_i = p_0 \frac{\lambda_d}{\beta} \frac{(1 - e^{-\gamma})^2}{\gamma} e^{i\gamma} \qquad (i = 2,3,\ldots) \tag{2.12}$$

and hence the mean data queue length:

$$\begin{aligned} \bar{D} &= \sum_{i=1}^{\infty} ip_i = -p_0 \frac{\lambda_d}{\beta}\left(1 - \frac{1}{\gamma}\right) \\ &= \frac{\rho_d}{v(1-\rho)}\left[1 + \frac{(1-\theta^2)/v^a+\theta^2}{\rho_d + v(1-\rho)} \cdot \frac{\mu_d}{\mu_v\rho_v} \sum_{i=0}^{v-1} \frac{T_i^2}{\pi_i}\right] \end{aligned} \tag{2.13}$$

Higher moments can also be readily derived. For the special case v=1, we have from (2.13)

$$\bar{D} = \frac{\rho_d}{1 - \rho}\left[1 + \frac{\mu_d}{\mu_v} \frac{\rho_v}{(1 + \rho_v)^2}\right] \tag{2.14}$$

which gives the exact result (see, e.g., [16]).

Numerical results from our model are compared with those of fluid approximation originally reported in [8] for a system with v=10, d=0, $\mu_d/\mu_v = 10^4$ and various combinations of ρ_v and ρ_d (fig. 1); and also with those of simulation originally reported in [13] for a system with v=10, d=5, $\mu_d/\mu_v = 10^4$, ρ_v=5 and ρ_d varying between 5 and 10 (fig. 2). In all cases studied our diffusion approximation model gives results with satisfactory accuracy.

3. TRANSIENT BEHAVIOUR

In order to study the transient behaviour of the data queue, we

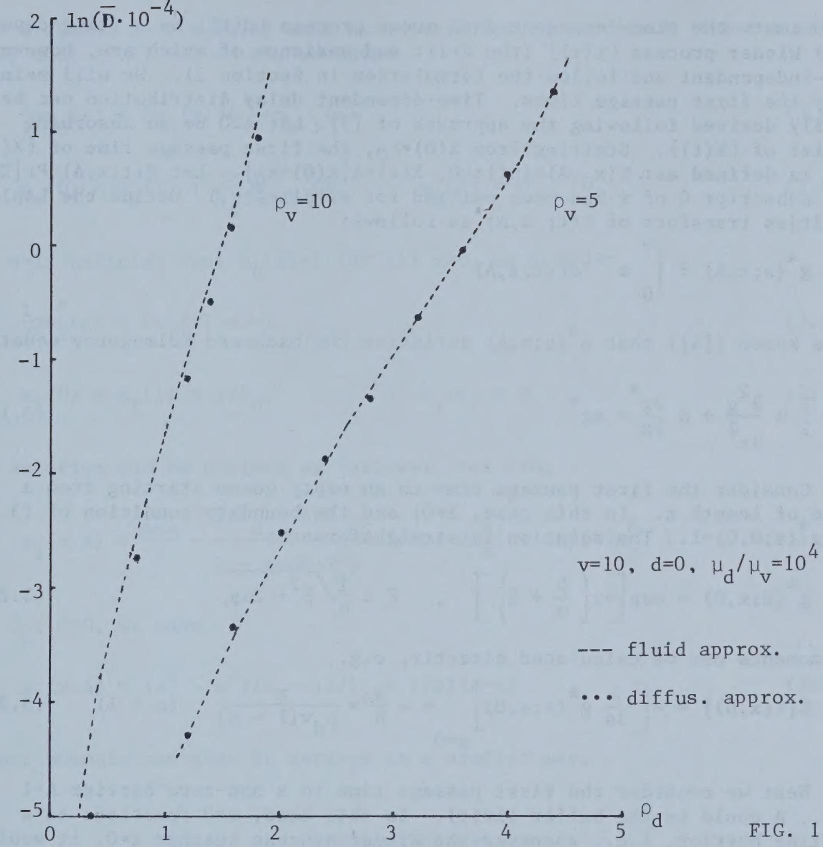

$\ln(\overline{D}\cdot 10^{-4})$
2
1
0
-1
-2
-3
-4
-5
$\rho_v=10$
$\rho_v=5$
$v=10,\ d=0,\ \mu_d/\mu_v=10^4$
--- fluid approx.
••• diffus. approx.
0
1
2
3
4
5
ρ_d

FIG. 1

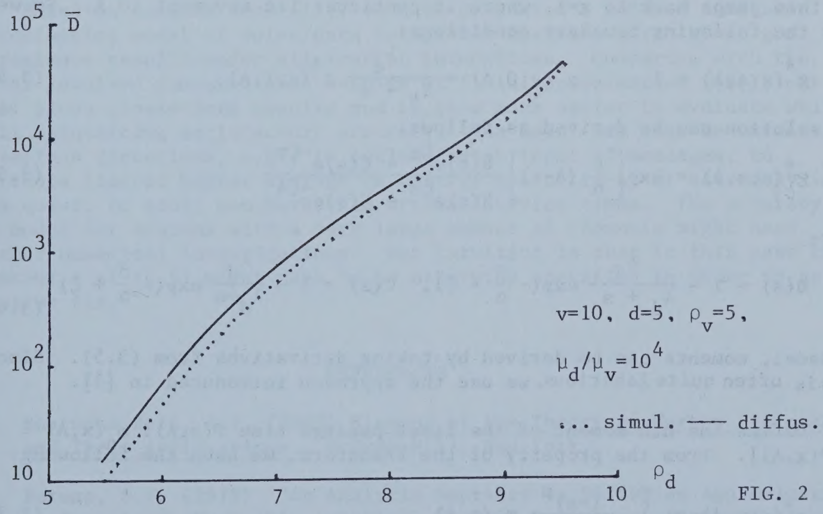

10^5 $\overline{D}$
10^4
10^3
10^2
10
$v=10,\ d=5,\ \rho_v=5,$
$\mu_d/\mu_v=10^4$
... simul. —— diffus.
5
6
7
8
9
10
ρ_d

FIG. 2

approximate the time-dependent data queue process $\{D(t)\}$ by a (time-dependent) Wiener process $\{X(t)\}$ (the drift and variance of which are, however, time-independent and follow the formulation in Section 2). We will mainly study the first passage times. Time-dependent delay distribution can be readily derived following the approach of [7]. Let $A \geq 0$ be an absorbing barrier of $\{X(t)\}$. Starting from $X(0)=x_0$, the first passage time of $\{X(t)\}$ to A is defined as: $T(x_0,A) \equiv \inf[t \geq 0,\ X(t)=A \mid X(0)=x_0]$. Let $P(t;x,A) \equiv \Pr[T \leq t]$. (The subscript 0 of x has been omitted for simplicity.) Define the Laplace-Stieltjes transform of $P(t; x,A)$ as follows:

$$g^*(s;x,A) \equiv \int_0^\infty e^{-st} dP(t;x,A)$$

It is known ([4]) that $g^*(s;x,A)$ satisfies the backward Kolmogorov equation:

$$\frac{1}{2}\alpha \frac{\partial^2 g}{\partial x^2} + \beta \frac{\partial g^*}{\partial x} = sg^* \tag{3.1}$$

Consider the first passage time to an empty queue starting from a queue of length x. In this case, A=0; and the boundary condition of (3.1) is: $g^*(s;0,0)=1$. The solution is straightforward:

$$g^*(s;x,0) = \exp\left[-x\left(\frac{\beta}{\alpha} + \xi\right)\right] \quad , \quad \xi \equiv \frac{1}{\alpha}\sqrt{\beta^2 + 2\alpha s} \tag{3.2}$$

And moments can be calculated directly, e.g.,

$$E[T(x,0)] = -\left[\frac{\partial}{\partial s} g^*(s;x,0)\right]_{s=0} = -\frac{x}{\beta} = \frac{x}{\mu_d v(1-\rho)} \qquad (\rho < 1) \tag{3.3}$$

Next we consider the first passage time to a non-zero barrier $A>1$ (e.g., A could be the buffer limit). In this case, $x=0$ functions as a visiting barrier, i.e., whenever the Wiener process reaches $x=0$, it would stay there for an interarrival time (the transform of which is $\lambda_d/(\lambda_d + s)$) and then jumps back to $x=1$, where it continues its movement to A. So we have the following boundary conditions:

$$g^*(s;A,A) = 1, \qquad g^*(s;0,A) = \frac{\lambda_d}{\lambda_d + s} g^*(s;1,A) \tag{3.4}$$

The solution can be derived as follows:

$$g^*(s;x,A) = \exp\left[\frac{\beta}{\alpha}(A-x)\right] \frac{B(s)e^{\xi x} - C(s)e^{-\xi x}}{B(s)e^{\xi A} - C(s)e^{-\xi A}} \tag{3.5}$$

where

$$B(s) = 1 - \frac{\lambda_d}{\lambda_d + s}\exp\left(-\frac{\beta}{\alpha} - \xi\right), \quad C(s) = 1 - \frac{\lambda_d}{\lambda_d + s}\exp\left(-\frac{\beta}{\alpha} + \xi\right) \tag{3.6}$$

As usual, moments can be derived by taking derivatives from (3.5). Since this is often quite laborious, we use the approach introduced in [4].

Define the nth moment of the first passage time $T(x,A)$: $m_n(x,A) \equiv E[T^n(x,A)]$. From the property of the transform, we have the following:

$$g^*(s;x,A) = \sum_{n=0}^{\infty} \frac{(-s)^n}{n!} m_n(x,A) \tag{3.7}$$

(The argument A of $m_n(x,A)$ will be omitted for simplicity.) Substituting (3.7) into (3.1) and (3.4), we have:

$$\frac{1}{2}\alpha m_n''(x) + \beta m_n'(x) = -n m_{n-1}(x) \tag{3.8}$$

$$m_n(0) = m_n(1) + \frac{n}{\lambda_d} m_{n-1}(0), \qquad m_n(A) = 0 \tag{3.9}$$

For n=1, noticing that $m_0(x)=1$ for all $x \geq 0$, we have

$$\frac{1}{2}\alpha m_1''(x) + \beta m_1'(x) = -1 \tag{3.10}$$

$$m_1(0) = m_1(1) + 1/\lambda_d, \qquad m_1(A) = 0 \tag{3.11}$$

The solution can be derived as follows: for $\beta \neq 0$,

$$m_1(x,A) = \frac{A-x}{\beta} - \frac{1/\lambda_d - 1/\beta}{1 - \exp(-2\frac{\beta}{\alpha})} \left[\exp(-2A\frac{\beta}{\alpha}) - \exp(-2x\frac{\beta}{\alpha})\right] \tag{3.12}$$

and for $\beta=0$, we have

$$m_1(x,A) = (A^2 - x^2)/\alpha - (1/\lambda_d - 1/\alpha)(A-x) \tag{3.13}$$

Higher moments can also be derived in a similar way.

4. CONCLUDING REMARKS

Based on results from [11], we have presented a refined diffusion approximation model of voice/data integrated systems, which gives good approximate results under all traffic intensities. Comparing with the rather involved computational results of fluid approximation ([8]), our model gives closed-form results and is thus much easier to evaluate while still maintaining satisfactory accuracy. The model can also be extended in various directions, e.g., to include batch input of messages, to introduce limited buffer storage to control the build-up of extremely long data queue, to study non-Markovian arrival/service times. The accuracy of the model for systems with a very large number of channels might need further numerical investigations. Our intuition is that in this case the constant a of (2.5) might need to be otherwise specified in order to get a better fit.

REFERENCES

[1] Bharucha-Reid, A.T. (1960) Element of the Theory of Markov Processes and Their Applications. McGraw-Hill, New York.

[2] Burman, D.Y. (1979) "An Analytic Appraoch to Diffusion Approximations in Queueing", Ph.D. Dissertation, Courant Institute of Mathematics, New York University.

[3] Cox, D.R. (1968) Renewal Theory, Methuen, London.

[4] Cox, D.R. and Miller, H.D. (1968) The Theory of Stochastic Processes. Wiley, New York.

[5] Disney, R.L. and Cherry, W.P. (1974) "Some Topics in Queueing Network Theory." In Mathematical Methods in Queueing Theory, A.B. Clarke (ed.), Springer-Verlag, Berlin.

[6] Feller, W. (1954) "Diffusion Processes in One Dimension." Trans. Am Math. Soc. 77, 1-31.

[7] Gaver, D.P. (1968), "Diffusion Approximations and Models for Certain Congestion Problems." J. Appl. Prob. 5, 607-623.

[8] Gaver, D.P. and Lechoczky, J.P. (1982) "Channels that Cooperatively Service a Data Stream and Voice Messages." IEEE Trans. on Communications, COM-30, 1153-1162

[9] Gelenbe, E. (1975) "On Approximate Computer System Models." JACM 22, 261-269.

[10] Gelenbe, E. and Mitrani, I. (1980) Analysis and Synthesis of Computer Systems. Academic Press, London.

[11] Lehoczky, J.P. and Gaver, D.P. (1981) "Diffusion Approximations for the Cooperative Service of Voice and Data Messages." J. Appl. Prob. 18, 660-671.

[12] Lehoczky, J.P. and Gaver, D.P. (1981) "Performance Evaluation of Voice/Data Queueing Systems." Tech. Report #191, Dept. of Statistics, Carnegie-Mellon University, Pittsburgh, PA 15213.

[13] Leon-Garcia, A., Kwong, R. and Williams, G. (1982) "Performance Evaluation Methods for an Integrated Voice/Data Link." Tech. Report, Dept. of Electrical Engineering, University of Toronto.

[14] Neuts, M.F. (1981) Matrix-Geometric Solutions in Stochastic Models --- An Algorithmic Approach. The Johns Hopkins University Press, Baltimore.

[15] Sevcik, K.C. et al. (1977) "Improving Approximations of Aggregated Queueing Network Subsystems." In Computer Performance, K.M. Chandy and M. Reiser (eds.), North Holland.

[16] Weinstein, C.J., Malpass, M.L. and Fischer, M.J. (1980) "Data Traffic Performance of an Integrated Circuit-and Packet-Switched Multiplex Structure." IEEE Trans. on Communications, COM-28, 973-978.

UPPER AND LOWER BOUNDS ON THE AVERAGE ERROR PROBABILITY FOR OPTICAL COMMUNICATION SYSTEMS

J.R.F. da Rocha and John J. O'Reilly
Department of Electrical Engineering Science
University of Essex, U.K.

SUMMARY

Recently Prabhu[1] has presented upper and lower bounds on the average error probability for binary baseband electrical communication. These bounds cannot be applied directly to optical communications due to the signal dependent character of the quantum noise contributions. However, we have previously shown[2] that an upper bound can be obtained for this case following broadly Prabhu's philosophy; we now report a lower bound and comment briefly on the application of both bounds to the performance evaluation of practical optical communication systems.

Conditional upper bounds are derived in [2] as:

$$P_{e_1} \leq Q_{u_1} = \frac{M_{Y_1}(-s)}{\sqrt{2\pi}\, s\sigma} \exp\{Fs + \sigma^2 s^2/2\}$$

$$P_{e_0} \leq Q_{u_0} = \frac{M_{Y_0}(s)}{\sqrt{2\pi}\, so} \exp\{-Fs + \sigma^2 s^2/2\}$$

where $M_{Y_i}(s)$ denote signal moment generating functions, σ is the rms level of the additive gaussian noise and F is the decision threshold. By a broadly similar, albeit rather more involved procedure, lower bounds on the symbol conditioned error probabilities can be obtained[3] as:

$$Q_{L_i} = Q_{u_i} \left\{ \frac{2}{1+Z} - s^2 A_i \frac{Z-1}{Z+1} - \frac{2}{\sqrt{2\pi}\, s\sigma} \right\}, \quad i = 0,1$$

where:

$$A_0 = \frac{1}{2M_{Y_0}(s)} \{ M''_{Y_0}(s) - 2\,(F - s\sigma^2)\, M'_{Y_0}(s) + (F - s\sigma^2)^2\, M_{Y_0}(s) \}$$

$$A_1 = \frac{1}{2M_{Y_1}(-s)} \{ M''_{Y_1}(-s) + 2(F + s\sigma^2)\, M'_{Y_1}(-s) + (F + s\sigma^2)^2\, M_{Y_1}(-s) \}$$

and: $$Z = \left[1 + \frac{4}{(s\sigma)^2} \right]^{1/2}$$

The bounds on the average error probability are then obtained by summing these conditional bounds weighted by the element probabilities.

In applying these bounds to optical communication systems we have observed the following: For the upper bound, channel dispersion and avalanche gain have some influence, but the dominant factor governing bound tightness is the noise ratio NR, defined as the ratio of 'signal dependent' to 'additive' noise. For noise ratios corresponding to receivers of practical interest, the new upper bound offers more than an order of magnitude improvement over the Chernoff bound. Even for noise ratios as high as 20dB, considerably greater than are encountered in practice, some improvement is obtained.

In the case of the lower bound the situation is more complex. Here too, though, the noise ratio provides a useful basis for comparison and we find that the lower bound is tight (upper/lower bound ratio ~ 2 or 3), provided the signal dependent noise does not dominate. However, the presence of avalanche gain considerably degrades the bound for high noise ratios. Nevertheless, even with significant avalanche gain (G ~20), the tightness provided by the lower bound is comparable to that obtained with the Chernoff bound, up to the point where signal dependent noise begins to dominate over additive noise contributions (NR ~0dB).

From these investigations we conclude that the bounds are usefully tight for direct detection optical communication receivers with parameter values of practical interest. In most cases the bounds provide an accurate and computationally efficient method of performance evaluation and the existence of both upper and lower bounds means that this is achieved with known precision.

ACKNOWLEDGMENTS

One of the authors (J.R.F. da Rocha) is grateful to the Gulbenkian Foundation and University of Aveiro (Portugal) for financial support in relation to the work reported here.

REFERENCES

1.Prabhu,V.K.:" Modified Chernoff bounds for PAM systems with noise and interference ", IEEE Trans. Information Theory, January 1982, pp 95-100

2.Rocha,J.R.F. da and O'Reilly, J.J.:" A modified Chernoff bound for binary optical communication", Electron. Lett. Vol. 18, No. 16, (August 1982).

3.Rocha, J.R.F. da and O'Reilly, J.J.:'Error probability bounds for optical communication ", Telecommunication Systems Group Report No. 176, Optical Commmunications Laboratory, University of Essex, August 1982.

CHARACTERIZATION OF A HIGH AVAILABLE DUPLEX LARGE SCALE SYSTEM BY CELL PARTITION

Fabrizio Lombardi
Department of Electrical Engineering/Computer Science
Texas Tech University
Lubbock, TX 79404, USA

and

Owen J. Davies
Department of Electrical and Electronic Engineering
University College London
London WC1, United Kingdom

ABSTRACT

In this paper the availability modelling of a large scale duplex system is presented. The original duplex is divided into stages and cells to allow local fault detection by an intelligent unit. The implications of fail safe operation of the intelligent unit, buffering control of the stages are analyzed as a direct improvement to the availability. The steady state availability is derived using the frequency balancing technique. This has been generalized to accommodate the diversity of repair and failure rates.

INTRODUCTION

In modern society growing dependence on computer systems has inevitably focused attention on their reliability and performance [1], especially from a system effectiveness characterization.

In applications where the correct operation at any point in time is of primary concern, the availability has been used as measure [2].

Availability is a function of a failure process (associated with the reliability) and a regenerative procedure (maintenance, repair, replacement, etc.) [3,4]. If from the regenerative process, the attainment of high availability can be solved by various procedures [5,6], as according to financial constraint [4], the reliability offers interesting alternatives. The most used scheme to improve the reliability of a system is the odd N replication of the non redundant module and the final voting at the output to mask potential failures (NMR) [7,8]. It has been proved [8] that NMR is most effective if voting is performed at a level lower than modular. By dividing a module into submodules and cascading functional equally submodules in stages, a higher reliability can be achieved if voting is performed locally at each stage output. This improvement is dependent on the submodules failure rates and on the number of stages. A threshold for the complexity of the voting hardware, as set by the system replication factor N, must be met for maximum reliability.

A similar technique can be applied to duplex systems. Duplex [3,9,10] is the technique to provide redundancy in the system by duplicating the resources (or modules). Being in most cases the most effective solution to fault tolerance, duplex has been extensively used for large and expensive computer systems [3], and/or for applications where the extra redundant hardware should be kept to a minimum [11]. Examples of duplex systems are found in telephone exchange control [3,12] and business transaction systems [13]. These systems are characterized by the attribute of large scale [14] as either, they are part of a dynamical interconnected network or, the complexity of their operations is such that for maintenance purposes they must be broken down into submodules. These submodules are used in a serial configuration for improved throughput (from a performance evaluation analysis, they are effectively working in a pipeline configuration). In both these cases, the identification of stages can be readily accomplished. The functional characteristics of the stages as well as the fault tolerant strategies used, are diversified. Various techniques can be used. Notorious examples of these developments can be found in the ESS series [3] by Bell Laboratories for telephone exchanges.

DUPLEX SYSTEM CHARACTERIZATION

The analyzed duplex system is characterized as (figure 1):

a) Every submodule in a given stage is logical and physical independent. This is extended to a functional independence such that no common mode exists. In hardware, this is identified by different failure rates for their different complexity. In software, program diversity is used [15]. The implication of physical independence is accomplished by the generalized rate of the regenerative process (for convenience only, this process will be referred to as maintenance) to account for the different submodule designs, the geographical separation from qualified staff [3] and the unavoidance in certain situations of a delay in the maintenance process [5].

b) Every submodule is provided with testing capabilities. These capabilities can be identified as the ability to self test and the ability to inter-submodule test with the other submodule in the stage. This inter-submodule test is performed on an independent and bidirectional T bus. The effect of coverage [16] (defined as the probability of identifying the attributes of a fault [17]) can be added to the failure and repair rates as a dumping factor, i.e. maintenance can be improved if a precise and fully covered characterization of the faults can be achieved. These testing capabilities can be either run concurrently with the on-line processing [18] and/or they can be considered as off-line activity (the applicability of one compared with the other is application dependent).

c) The flow of data in the network is allowed in unidirectional and independent links. All the submodules along the flow of data are referred to as a channel.

d) At every stage an interface unit, referred to as the intelligent unit (IU) [19], is provided. This unit is used for on-line fault detection such that on the occurrance of a disagreement in the response of the submodules, off line test (i.e., self and intersubmodule test) can be initialized. The IU has built-in fail-safe logic [20] such that its failure is not catastrophic. The IU provides the facility of link switching between the two channels.

e) Failure of a given submodule is not catastrophic for the unidirectional link. This can be accomplished by proper buffering as enabled by the IU.

f) Link switching is negligible to system operation.

g) Coupling of stages is possible by the link switching procedure and by the self-contained nature of the processing of each submodule. No logical conflict is encountered in the switching of the links, i.e. operation of a stage is totally safeguarded by the failure of the precedent stage.

As for the system shown in figure 1, a cell can be defined as a subdivision made of a full stage (i.e. two submodules and associated IU) and the next two submodules in the sequential stage. This division is required in order to take into account the effects of the link switching as provided by the IU and the effect of concurrency in the testing (i.e. the first two submodules in a cell are on line tested by the second two submodules, by concurrent software detection methods [21,22].

CELL ANALYSIS

A cell, as previously defined, can be visualized as a star network [23,24]. In [25,26], a star network is analyzed using a cut set approach. However, due to the large scale system characterization (the number of stages can be very large such that the combinations grow to a vast number) and the requirement of particularity for both the failure and repair rates of the submodules and the IUs, the cut set approach is rather cumbersome, both in term of computer time and memory requirements. This can be circumvented by using the frequency balancing technique [26] for the solution of the steady state availability of the cells. However, this requires the formulation of a generalized procedure for the cell availability and then to establish the relationships between the cascading cells to calculate the system availability.

The state transition diagram of a cell is shown in figure 2 (for simplicity, only the failure transitions are indicated). The failure and the repair rates (λ_i and μ_i respectively, where the subscript refers to the position number as in figure 1) are exponentially distributed. The states in the transition diagram are denoted by S_{ij}. The first subscript, i, refers to the hierarchical state level as caused by the number of faulty elements in the cell (i.e. for two faulty elements, the state is referred to as 3rd hierarchical to account for the 1st hierarchical state of initial fault free operation). The second subscript, j, specifies the faulty elements in the cell (i.e. for j = (1,2,4), it implies that submodules 1,2 and 4 are faulty; note that the order of the faulty processors is not important). This is very useful in providing a descriptive notation to the status of the system.

The first state equation which refers to the first hierarchical level can be written as

$$P_{1,0}\,(\lambda_1 + \lambda_2 + \lambda_3 + \lambda_4 + \lambda_5) = \sum_{\alpha=1}^{5} P_{2,\alpha}\,\mu_\alpha \qquad (1)$$

where P denotes the probability of being in the state as specified by the subscripts.

For the secondary states, the equations are of the forms

$$P_{2,\alpha}\,(\mu_\alpha + \sum_{\substack{\beta=1\\ \beta\neq\alpha}}^{5} \lambda_\beta) = \lambda_\alpha P_{1,0} + \mu_\alpha \sum_{\substack{\beta=1\\ \beta\neq\alpha}}^{5} P_{3,(\alpha,\beta)} \qquad (2)$$

for all α, $\alpha = 1,5$

The third and fourth state equations are given respectively by

$$P_{3,(\alpha,\beta)}(\sum_{\substack{\gamma=1\\ \gamma\neq\alpha\neq\beta}}^{5} \lambda_\gamma + \mu_\alpha + \mu_\beta) = \lambda_\alpha P_{2,\alpha} + \lambda_\beta P_{2,\beta} + \sum_{\substack{\gamma=1\\ \gamma\neq\alpha\neq\beta}}^{5} \mu_\gamma P_{4,(\alpha,\beta,\gamma)} \qquad (3)$$

for all α and β, $\alpha \neq \beta$

and

$$P_{4,(\alpha,\beta,\gamma)}\Big(\sum_{\substack{\eta=1\\ \eta\neq\alpha\neq\beta\neq\gamma}}^{5} \lambda_\eta + \mu_\alpha + \mu_\beta + \mu_\gamma\Big) = \lambda_\alpha P_{3,(\beta,\gamma)} + \lambda_\beta P_{3,(\alpha,\gamma)}$$

$$+ \lambda_\gamma P_{3,(\alpha,\beta)} + \sum_{\substack{\eta=1\\ \eta\neq\alpha\neq\beta\neq\gamma}}^{5} \mu_\eta P_{5,(\alpha,\beta,\gamma,\eta)} \qquad (4)$$

for all α, β and γ $\alpha\neq\beta\neq\gamma$.

The fifth hierarchical level states are identified by the following equations

$$P_{5,(\alpha,\beta,\gamma,\eta)}\Big(\lambda_\phi + \sum_{\substack{K=1\\ K\neq\phi}}^{5} \mu_K\Big) = \mu_\phi P_{6,(\alpha,\beta,\gamma,\eta,\phi)} + \sum_{\substack{K=1\\ K\neq\phi}}^{5} \lambda_K P_{4,(\pi,\psi,\Omega)} \qquad (5)$$

where π,ψ,Ω are the three values of α,β,γ and such that $\phi\neq\alpha$ or $\phi\neq\beta$ or $\phi\neq\gamma$.

For the final and single sixth level state the equation is given by

$$P_{6,(\alpha,\beta,\gamma,\eta,\phi)} \sum_{K=1}^{5} \mu_K = \lambda_\alpha P_{5,(\beta,\gamma,\eta,\phi)} + \lambda_\beta P_{5,(\alpha,\gamma,\eta,\phi)} + \lambda_\gamma P_{5,(\alpha,\beta,\eta,\phi)}$$

$$+ \lambda_\eta P_{5,(\alpha,\beta,\gamma,\phi)} + \lambda_\phi P_{5,(\alpha,\beta,\gamma,\eta)} \qquad (6)$$

for $\alpha\neq\beta\neq\gamma\neq\eta\neq\phi$

The above 32 equations can be solved by recursive formulation of α,β,γ,η and ϕ terms and using the total expression

$$P_{1,0} + \sum C\left[P_2\right] + \sum C\left[P_3\right] + \sum C\left[P_4\right] + \sum C\left[P_5\right] + P_{6,(1,2,3,4,5)} = 1 \qquad (7)$$

where $C\left[P_i\right]$ represents the combinations of possible fault patterns for the hierarchical level i (i=2,5)

By extensive mathematical manipulations, the following solutions have been found:

a) First hierarchical:

$$P_{1,0} = 1/Z \qquad (8)$$

b) Second hierarchical:

$$P_{2,i} = X_i/Z \quad i=1,5 \qquad (9)$$

c) Third hierarchical:

$$P_{3,(i,J)} = X_i X_J / Z \qquad i,J=1,5 \qquad i \neq J \tag{10}$$

d) Fourth hierarchical:

$$P_{4,(i,J,K)} = X_i X_J X_K / Z \qquad i,J,K = 1,5 \qquad i \neq J \neq K \tag{11}$$

e) Fifth hierarchical:

$$P_{5,(i,J,K,H)} = X_i X_J X_K X_H / Z \qquad i,J,K,H = 1,5 \qquad i \neq J \neq K \neq H \tag{12}$$

f) Sixth hierarchical

$$P_{6,(1,2,3,4,5)} = X_1 X_2 X_3 X_4 X_5 / Z \tag{13}$$

where

$$X_i = \lambda_i / \mu_i \qquad i = 1,5 \tag{14}$$

$$Z = X_1X_2X_3X_4X_5 + X_1X_2X_3X_4 + X_1X_2X_3X_5 + X_1X_2X_4X_5 + X_1X_3X_4X_5 + X_2X_3X_4X_5 + \\ + \sum_{\substack{i,J,K=1 \\ i \neq J \neq K}}^{5} X_u X_J X_K + \sum_{\substack{i,J=1 \\ i \neq J}}^{5} X_i X_J + X_1 + X_2 + X_3 + X_4 + X_5 + 1 \tag{15}$$

The availability of a cell is defined as the probability that at least one channel or both channels will be able to deliver the result of their computation (note that the computation can be performed partially in both channels). This means that data should be available either at point X or Y as in figure 1. The availability is therefore given by

$$A_C = P_{1,0} + \sum_{i=1}^{5} P_{2,i} + \sum_{i=3}^{5} P_{3,(1,i)} + \sum_{i=3}^{5} P_{3,(2,i)} + \sum_{i=3}^{4} P_{3,(i,5)} + \\ + P_{4,(1,3,5)} + P_{4,(2,4,5)} = P_X + P_Y \tag{16}$$

Note that the available states of a cell are shown in figure 2 as squares. The figure of the availability is decreased if either the IU is not fail safe or the two initial submodules in the cell are not properly buffered. In these cases the availability is given respectively by

$$A' = P_{1,0} + \sum_{\substack{i=1 \\ i \neq 5}}^{5} P_{2,i} + \sum_{i=3}^{4} P_{3,(1,i)} + \sum_{i=3}^{4} P_{3,(2,i)} \tag{17}$$

and

$$A'' = P_{1,0} + \sum_{\substack{i=1 \\ i \neq 5}}^{5} P_{2,i} + P_{3,(1,3)} + P_{3,(2,4)} \tag{18}$$

SYSTEM AVAILABILITY

The result of the cell availability can be exploited to find the duplex system availability. A serial model can be applied. However, some precautions must be set to allow for routing between cells as allowed by the intermediate switch. This is equivalent to transposing every cell as the first cell in the system and them taking into account the effect of the availability of the switch. For two adjoint cells (defined as C1 and C2), this means that the total availability is given by

$$A_T = A_{C1} \cdot A_{C2} \tag{19}$$

where

$$A_{C1} = \tag{20}$$

$$= A_1(1-P_W) + A_1 P_W = A_1$$

and P_W is the availability of the output switch, i.e.,submodule 6 in Figure 1. (This switch is assumed to be fail safe). If 6 is not fail safe then

$$A_{C1} = A_1 \ P_W \tag{21}$$

In general for a duples system made of N cells (denoted as C_1, C_2, C_3, ... C_N), N being an even number, the availability is given by

$$A_S = \overset{N}{\underset{i=1}{\pi}} A_{Ci} \tag{22}$$

CONCLUSIONS

A method for calculating the availability of large scale duplex systems has been presented. This is based on the frequency balancing approach. This method offers the advantage of simplicity as compared to a cut set approach and it can be readily implemented for computer evaluation. Future research is pursued in order to characterize complex networks with expansion in the number of channels and to include different redundancy characteristics (TMR and NMR). The possibility to apply this method to testing [27,28] is being currently explored.

REFERENCES

[1] Avizienis, A. "Fault Tolerant Computing: Progress, Problems and Prospects" Proc. IFIP Congress, pp. 405-420, Toronto, 1977.

[2] Randell, B., and T. Anderson. "Computing System Reliability" Cambridge University Press, 1979.

[3] Kraft, G.D., and W.N. Toy, "Microprogrammed Control and Reliable Design of Small Computers" Prentice Hall, N.J., 1981.

[4] Lombardi F., and O.J. Davies "Availability Analysis of Fault Tolerant Microcomputer Systems with Periodic Preventive Replacement" Proc. 8th Symp. of the European Association of Microprocessing and Microprogramming, North Holland Publ. Co., Haifa, 1982 (to appear).

[5] Lombardi, F. "Analysis and Modelling of a N Modules Parallel System with Restructuring Capabilities under Permanent and Transient Faults" submitted for publication to Performance Evaluation: an International Journal, North Holland Publ. Co.

[6] Lombardi, F. "On the Imperfection of Replacement" Microelectronics and Reliability, Pergamon Press, Oxford, in press.

[7] Siewiorek, D.P. "Reliability Modelling of Compensating Module Failures in Majority Voted Redundancy" IEEE Trans. on Comp., Vol. C24, No. 5, pp 525-533, 1975.

[8] Lyons, R.E., and W. Vanderkulk, "The Use of Triple Modular Redundancy to Improve Computer Reliability" IBM Journal of Research and Development, pp. 200-209, 1962.

[9] Sedmak, R.M., and H.L. Liebergot "Fault Tolerance of a General Purpose Computer Implemented by Very Large Scale Integration" IEEE Trans. on Comp., Vol. C29, No. 6, pp. 492-500, 1980.

[10] Obac Roda, V. and F. Lombardi, "Software Implementation of a Disagreement Detector for a Duplex Hybrid System" Eurocon 82, Electrotechnics Conf., pp. 673-678, Proc. by North Holland Publ. Co., edited by Langer and Multoft, 1982.

[11] Avizienis, A. "Architecture of Fault Tolerant Computing Systems" FTCS, pp. 3-16, 1975.

[12] Bellman, A. "Redundancy Design Approach Applied to Electronic Telephone Exchange Realization" FTCS, pp. 9-16, 1976.

[13] Tandem Comp. Inc. "The Non-Stop Systems" Cupertino, 1982.

[14] Saeks, R., and R. DeCarlo, "Interconnected Dynamical System" Marcel Dekker, N.Y. 1981.

[15] Wulf, W.A., "Reliable Hardware/Software Architecture" IEEE Trans. on Soft., Eng., Vol. SE1, No. 2, pp. 233-240, 1975.

[16] Arnold, T.F. "The concept of Coverage and its Effect on the Reliability Model of a Repairable System" IEEE Trans. on Comp., Vol. C22, No. 3, pp. 251-254, 1973.

[17] Avizienis, A., "The Four Universe Information System Model for the Study of Fault Tolerance" FTCS, pp 6-13, 1982.

[18] Ramamoorthy, C.V., and Y.W. Han, "Reliability Analysis of Systems with Concurrent Error Detection" IEEE Trans. on Comp., Vol. C24, No. 9, pp. 868-878, 1975.

[19] Lombardi, F., Obac Roda, V. and Md. M. Islam, "Reliability Study of Duplex Hybrid Systems" Microelectronics and Reliability, Vol. 22, No. 3, pp 457-471, Pergamon Press, Oxford, 1982.

[20] Siewiorek, D.P. and R.S. Swarz, "The Theory and Practice of Reliable System Design" Digital Press, Bedford, 1982.

[21] Lombardi, F., "Microcomputer Real Time Software Reliability and Fault Recovery" Microelectronics and Reliability, Pergamon Press, Oxford, in press.

[22] Kane, J.K. and S.S. Yau, "Concurrent Software Fault Detection" IEEE Trans. on Soft. Eng., Vol. SE 1, No. 1, pp. 87-99, 1975.

[23] Gupta, H. and J. Sharma, "A Delta Star Transformation Approach for Reliability Evaluation" IEEE Trans. on Rel. Vol. R27, No. 3, pp. 212-214, 1978.

[24] Banerjee, J.K. and K. Rajamani, "Closed Form Solutions for Delta-Star and Star-Delta Conversions for Reliability Networks" IEEE Trans. on Rel., Vol. R25, No. 2, pp 118-119, 1976.

[25] Singh, C. and M.D, Kankam, "Correspondence: Comments on Closed Forms Solutions for Delta-Star and Star-Delta Conversions for Reliability Networks" IEEE Trans. on Rel., Vol. R25, No. 12, pp. 336-339, 1976.

[26] Singh, C. and R. Billington, "System Reliability Modelling and Evaluation" Hutchinson, London, 1977.

[27] Feng, T.Y. and C.L. Wu, "Fault Diagnosis for a Class of Multistage Interconnection Networks" IEEE Trans. on Comp., Vol C23, No. 10, pp. 743-758, 1981.

[28] Lombardi, F., and O.J. Davies, "System Diagnosability Using Triplet Assertion" IEEE Int. Symp. on Large Scale Systems, Virginia Beach, October 1982 (to appear).

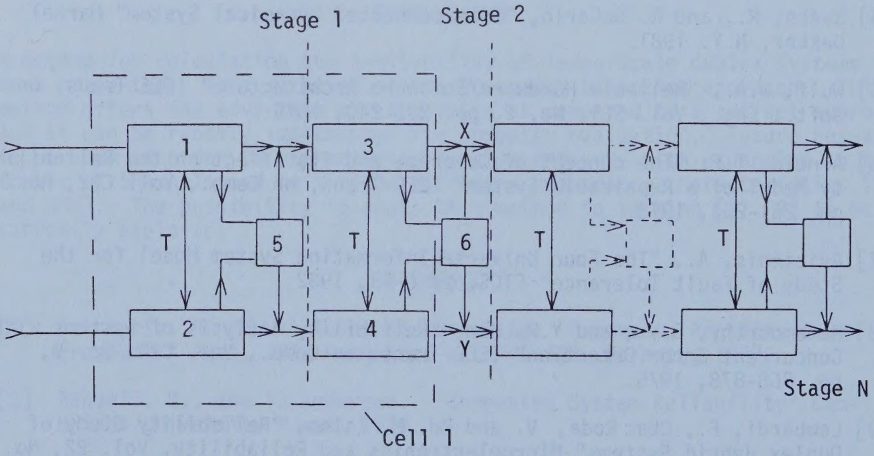


Figure 1. Duplex System

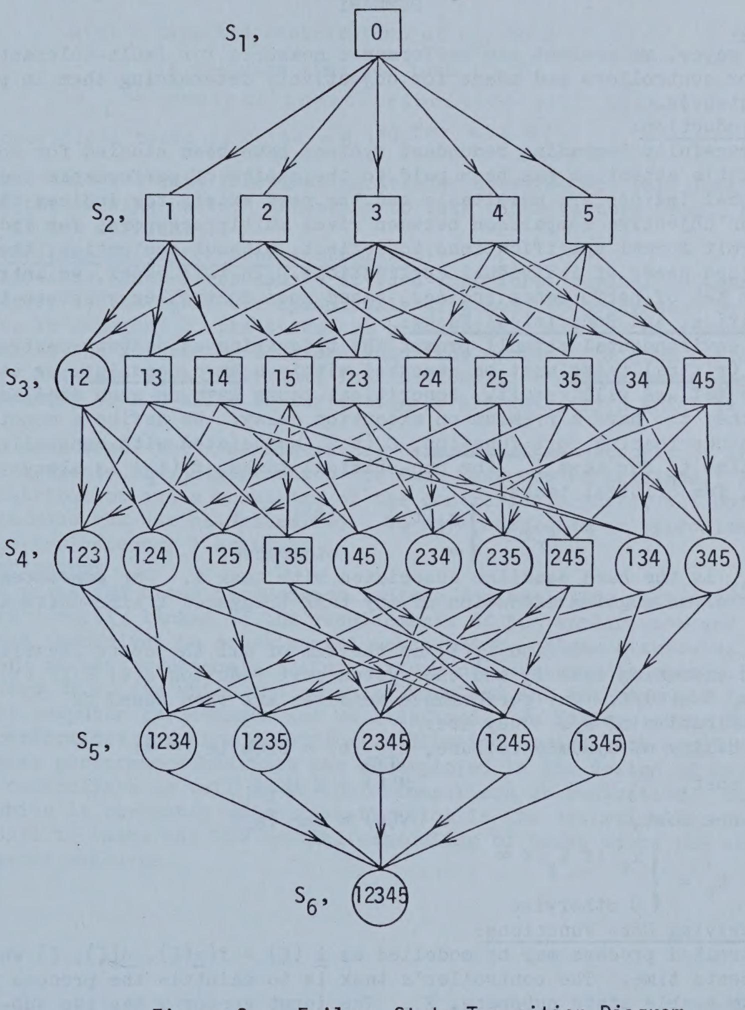

S1, 0
S2, 1 2 3 4 5
S3, 12 13 14 15 23 24 25 35 34 45
S4, 123 124 125 135 145 234 235 245 134 345
S5, 1234 1235 2345 1245 1345
S6, 12345

Figure 2. Failure State Transition Diagram

PERFORMANCE MEASURES FOR FAULT-TOLERANT MULTIPROCESSOR CONTROLLERS FOR CRITICAL PROCESSES*

C.M. KRISHNA AND K.G. SHIN
Department of Electrical and Computer Engineering,
The University of Michigan,
Ann Arbor, MI 48109

SUMMARY

Abstract:

In this paper, we present new performance measures for fault-tolerant multiprocessor controllers and means for objectively determining them in practical instances.

I. Introduction:

While gracefully degrading redundant systems have been studied for some time, little attention has been paid to the choice of performance indices. Traditional indices are unsuitable and the need exists for indices that permit an objective comparison between rival multiprocessors, for indices that permit formal specifications to reflect, without contortion, the specialized needs of individual applications. In this paper, we introduce one such set of performance indices, based upon controller response time.

II. Critical and Noncritical Tasks:

Certain environmental stimuli prompt the triggering of certain control tasks. Critical tasks must be executed within a hard deadline, or catastrophic failure will result. Noncritical tasks have no such deadline although they too have a premium on execution delay. We define a monotonically nondecreasing cost function, $C_i(t_r)$, associated with controller response time t_r for task i. For non-critical tasks, $C_i(t_r)$ is always bounded; for critical tasks,

$$C_i(t_r) = \begin{cases} g_i(t_r) & \text{if } t_r < t_{di} \\ \infty & \text{otherwise} \end{cases}$$

where t_{di} is the hard deadline associated with task i. For consistency, the cost of an ongoing execution of any task triggered τ time-units ago is $C_i(\tau)$.

Define $\Gamma_i(t)$ as the sum of all the costs incurred in $[0,t]$ in executing task i, and the system cost function $s(t) \equiv \sum_i \Gamma_i(t)$. Define $k_t \equiv s(t)/t$. Our performance measures are then based on the distribution of k_t. They are:

Probability of dynamic failure, $p_{dyn}(t) = \text{Prob}\,\{k_t = \infty\}$

Mean cost, $M(t) = E\,\{k_t'\}$

Variance cost, $V(t) = \text{var}\,\{k_t'\}$

where

$$k_t' = \begin{cases} k_t & \text{if } k_t < \infty \\ 0 & \text{otherwise} \end{cases}$$

III. Deriving Cost Functions:

The controlled process may be modelled as $\underline{\dot{x}}(\xi) = f(\underline{x}(\xi), \underline{u}(\xi), \xi)$ where ξ represents time. The controller's task is to maintain the process within an admissable state subspace, $\underline{X}_A$. The input vector $\underline{u}$ has two subcomponents: the environmental input $\underline{u}_e$ and the controller input $\underline{u}_c$. $\underline{u}_c$ is a function of $\underline{u}_e$, $\underline{X}_A$, current state, and response time. $\underline{X}_A$ is prespecified and the environment is assumed to be stochastically known. Physical limitations bound $\underline{u}_c$. With this information, the deadlines t_{di}

---

*This work is supported in part by NASA grant No. NAG 1-296.

can be obtained.

The traditional formulation of the instantaneous performance index for a controlled process is $f_0(\xi) = g(\underline{x}(\xi), \underline{u}(\xi), \xi)$. Overall "cost" of running the process is given by $J = \int_0^T f_0(\xi)\, d\xi$, where T is the "mission lifetime". We claim that a good representation for $g_i(t_r)$ is:

$$g_i(t_r) = \Omega(t_r) - \Omega(0)$$

where $\Omega(\eta)$ = Expected contribution of u_{ci} to J if response time for task i = η

u_{ci} = control input associated with task i.

(For noncritical tasks j, $c_j(t) = g_i(t)$ for all t).

Note that the cost functions as defined above do not take into account any but the running costs. Extension to take other factors (e.g., hardware costs) is not difficult.

IV. Modelling Multiprocessor Controllers:

Multiprocessor controllers can be divided into three logical categories. In category 1, the tasks are dynamically allocated to processors by a dispatcher; in category 2, the tasks are pre-assigned and act as execution triggers to the relevant processors when they arrive. (A good analogy is dataflow machines with individual instructions in the dataflow programs corresponding to sub-tasks in our analogy); and category 3 which is a hybrid.

In each case, the controllers may be modelled as networks of queues and the response time distribution for each task can be obtained. From this distribution and a knowledge of the controlled process, all our performance measures can be determined. We have completed an approximate solution for category 1 systems.

V. Conclusions

We have presented above performance indices for real-time controllers (i) that are directly linked to the requirements of the application and (ii) that lend themselves to experimental measurement. These attributes, taken together, result in uniquely useful performance measures in specialized applications, i.e., in which the computer workload and the economic implications of computer performance are well-defined. We believe these to be the first performance measures specifically directed toward such applications.

These performance measures can be employed in the design of multiprocessor controllers as well as in their comparison or evaluation. One example which is currently under investigation is the optimal allocation of processors to tasks and the optimal scheduling of tasks using the above performance measures.

A PROPOSAL FOR FAULT-TOLERANCE IN A MULTIPROCESSOR SYSTEM

P. CORSINI

Istituto di Elettronica e Telecomunicazioni
Università di Pisa, Pisa, Italy

L. LOPRIORE

Istituto di Elaborazione dell'Informazione
Consiglio Nazionale delle Ricerche, Pisa, Italy

L. STRIGINI

Selenia, Industrie Elettroniche Associate S.p.A., Roma, Italy

In this paper, a fault-handling approach for distributed architectures is presented by referring to a specific multimicroprocessor system, i. e., the MuTEAM prototype. The approach is intended to take into account the following hardware and software specifications: i) the system is not provided with mass storage devices; ii) its activity consists of running a set of concurrent processes, cooperating through message exchanges; and iii) no erroneous output is generated toward the external environment even when a fault is present.

INTRODUCTION

Fault-handling strategies for multiprocessor systems may be roughly divided into two classes, tailored to two different models for system architecture. In the first model the architecture is physically partitioned into processors and mass storage devices. Each processor is subject to crash: however, a crashing processor may only corrupt the data involved in the device access possibly in progress. Moreover, erroneous system outputs possibly generated by that processor are tolerated by the external environment. After repair, the processor resumes operations using the state information available in mass storage. In its turn, a mass storage device may fail, which implies the loss of all data stored in that device: however, a device failure causes no harm to the contents of other devices. In these kinds of architecture, fault-tolerance is typically pursued either by stable storage methods /1/ or other consistency techniques /2/. Typical applications concern scientific environments, data-base management and personal computing.

In the architecture in the second model, there is no inherent separation between the process working areas and the state information relevant to recovery. A low-end example is a microprocessor-based controller made up of few tightly interacting components, so that a fault in one of them causes both a processor crash and the loss of state information. A high-end example is a multiprocessor system in a critical environment: here, the presence of interactions among processors outside mass storage makes it possible for a processor error to cause unacceptable disruption of operations in other processors. Fault-tolerance is usually pursued either by software replication /3,4/, or by system-wide ad hoc hardware /5,6/.

In this paper, we concern ourselves with multimicroprocessor distributed system architectures for real-time applications, and we present an alternative fault-handling strategy. The approach adopted was intended to take into account the following hardware and software characteristics:

1) The system does not need to be provided with mass storage devices. This implies that the amount of recovery information stored must be as small as possible. Moreover, the storage medium is volatile, and a failure in one processor may imply the loss of the whole contents of the associated memory banks. Processors are subject to non-crash faults: no protection is enforced between processors and the storage containing system state information by the physical separation of the respective hardware environments.

2) The system must not generate erroneous outputs towards its external environment even when a fault is present. Moreover, the hardware is not provided with system-wide fault-handling facilities, nor is the application software replicated.
3) The software model does not define passive object entities, but only active processes whose global states represent the state of the system.

The fault-handling strategy is described in reference to a specific system, i.e., the MuTEAM prototype developed in Pisa /7/. The MuTEAM architecture /8/ features a clusterized /9-12/ multiprocessor organization. The clusters are loosely connected via serial links: the link topology is application specific, but must satisfy the constraint that each pair of nodes is connected by at least one communication path despite any single link failure. Each cluster (Fig. 1) is a set of one or more nodes, made up, essentially, of a processor, a shared memory bank (accessible to the processors of all the nodes in the same cluster) and a private memory bank (accessible only to the processor of the same node). In particular cases, a node may also be simply constituted by an intelligent peripheral. The MuTEAM system is provided with a kernel /13/ representing the run-time support for a message-passing concurrent language based on CSP /14/. The activity of the system consists of running a set of cooperating processes, distributed among the clusters and, in a given cluster, among the nodes constituting that cluster.

REQUIREMENTS FOR FAULT-TOLERANCE

Our requirement for system fault tolerance has been that the system must tolerate any single fault.

By single fault we mean: i) a node fault, i.e., one or more faults localized in the same node; or ii) any set of simultaneous node faults on nodes each of which belongs to a different cluster; or iii) a link fault, i.e., any set of simultaneous faults on intercluster

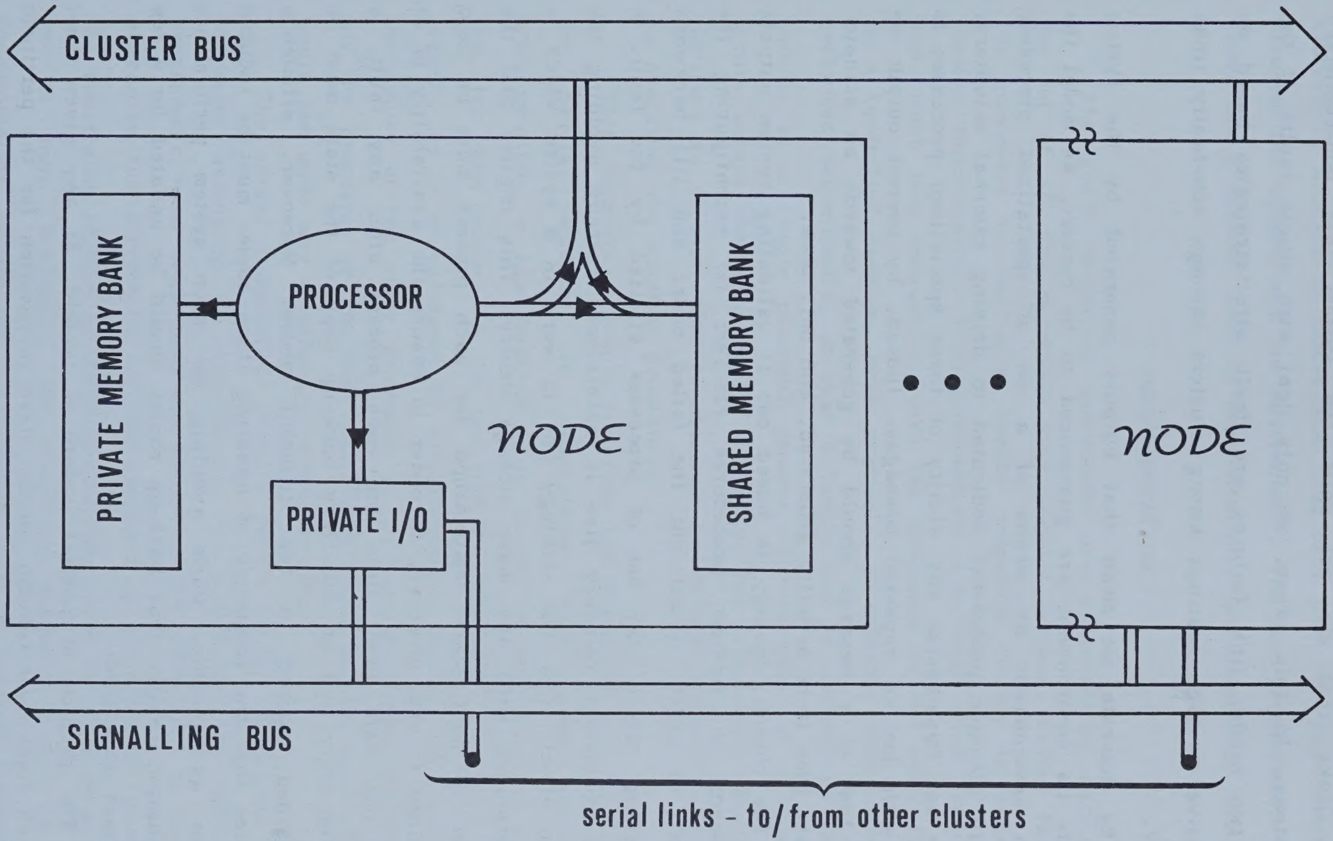

CLUSTER BUS
PRIVATE MEMORY BANK
PROCESSOR
PRIVATE I/O
NODE
SHARED MEMORY BANK
NODE
SIGNALLING BUS
serial links - to/from other clusters

Fig. 1. Configuration of a cluster of the MuTEAM system

serial links that do not split the system into non-connected subsystems. In this paper, we only deal with single faults of the first two kinds: link failures are dealt with strategies based on successive message routings among clusters through non-faulty links /15-17/.

By tolerate we mean that outputs generated by the system towards its environment are guaranteed to be correct. We model the system environment by means of a set of specialized processes (namely actuator processes) dedicated to driving external actuators. We do not hypothesize any ability of these specialized processes to deal with lost or repeated messages. Indeed, by correct output we mean that if a message should be generated towards an actuator process, then it is actually generated, and only once.

The chosen strategy is based on: i) validating system outputs by means of autotest procedures /18-19/; ii) reconfiguring the system, in order to cut out the failed node; and iii) backward recovering /20-21/ the set of processes affected by the fault. A single consistent recovery line is maintained, thereby avoiding the domino effect /22/. The strategy has to work in a system which is not provided with any mass storage device. This implies that the number of past states maintained for each process must be kept minimized to one. However, in order to ensure the availability of at least one safe past state for each process after any fault, a back-up copy and an auxiliary back-up copy of the state must be maintained, stored in two different nodes. Moreover, efficiency requires that the frequency of updating these copies must be reduced as far as possible, while avoiding too much system performance degradation implies that back-up copies should be updated for a few processes at a time.

The problem of partial update is twofold. At any given time: i) each copy must contain correct state information for the pertinent process; and ii) the entire contents of all the copies must be consistent, i.e., they must represent a state of the whole system allowed in absence of faults and congruent with the past interaction between the system and its environment. The two requirements stated above are a central issue in fault-tolerance for distributed systems.

IMPLEMENTATION

The goals mentioned previously are achieved by partitioning the set of all the processes into disjoint subsets, called families. The processes in the same family cooperate closely to achieve a common functionality. They communicate with each other frequently, and, much less frequently, with the processes of the other families in a controlled fashion. More precisely, interfamily interactions are performed sequentially, so that a family can be assimilated from its outside to a single powerful process. Furthermore, if a data flow occurs across the boundary of a family, the information involved is certified as correct: this is obtained by validating such information by executing the autotest procedures.

This suggests a simple criterion satisfying the two requirements for partial update: in coincidence with each interfamily communication, a new back-up copy is put in the place of the old one, for each process in the two families involved. This copy is: i) correct, as it is validated by autodiagnosis; and ii) congruent with the copies of the other processes in the system, since rolling back all the processes of the two families involved to this back-up state would not undo any interfamily communication.

The actual composition of each family (i.e., the names of the processes belonging to that family) is stated by the applicative programmer, and represents an input for the compiler. In order to guarantee the correctness of the messages to the actuator processes, the constraint should be followed whereby each of them is a family on its own. Moreover, the resulting frequency of interfamily communication should be the outcome from a tradeoff between two opposite requirements: i) the system must not be overburdened with too frequent activities of back-up state updating; and ii) the recovery line available in case of failure must not be too remote from the present state of the system.

In order to guarantee the required serialization of interfamily communications performed by processes belonging to the same given family, each family is provided with an arbiter process. Before starting an interfamily communication, each process must obtain a grant from the arbiter of its own family. This grant is returned only after the arbiter has communicated properly with the other arbiters, so as to avoid the enforced serialization of communications becoming a source of deadlock.

Arbiters are also responsible for starting a diagnostic and fault-handling (DFH) session in coincidence with each interfamily communication. Indeed, a request from one single arbiter is sufficient to trigger off a DFH session: proper software mechanisms embedded in DFH algorithms guarantee that multiple requests for system diagnosis issued during a DFH session do not cause useless repetition of the session. In order to ensure that at least one arbiter starts a DFH session at each given interfamily communication even when a faulty node is present, arbiters of different communicating families are allocated in different nodes. Moreover, a further restriction is that any two communicating processes belonging to different families must be allocated in different nodes. In such a way, we are assured that at least one of the two processes performs a request for interfamily communication to its arbiter.

After a fault has occurred and has been diagnosed in node N_f, proper reconfiguration activities are carried out reallocating all the processes previously allocated in N_f to non-faulty nodes. In actual fact, in order to avoid possible inefficiencies in the resulting workload distribution among nodes, we provide a mechanism which allows the reallocation of the processes allocated in every node. More precisely, according to our model for faults, in each cluster as many different faults may occur as there are nodes in that cluster: for each fault, we store proper information about where to reallocate every process in the cluster. However, efficiency considerations for reconfiguration suggest that, in order to generate the lowest information flow: i) a process allocated on a non-faulty node should be reallocated on the node containing its back-up copy, or in its own node (storing the process auxiliary back-up copy); and ii) a process allocated on the faulty node should be reallocated on the node containing its back-up copy. These criteria should also be kept in mind when distributing back-up copies and auxiliary back-up copies among nodes.

CONCLUSIONS

A method for fault-handling has been presented, designed for multiprocessor systems supporting concurrent processes cooperating through message exchange. The proposal has been described in reference to a specific system, i.e., the MuTEAM prototype developed in Pisa: our requirement was that no erroneous output be generated by the system under a single fault hypothesis. Fault-handling is based on backward error recovery: the set of all the application processes is partitioned into disjoint subsets (called families), which represent the atomic unit of recovery. Recovery points are established on communications among families. A single consistent recovery line is maintained, thereby avoiding the domino effect. The model does not rely on the usage of mass storage devices: rather, the recovery information pertinent to all the processes is kept in the distributed main memory of the system.

Further details about the implementation of these fault-handling stategies on the MuTEAM system can be found in /23/.

ACKNOWLEDGEMENT

This work has been supported by the National Computer Science

Program of the Italian National Research Council.

REFERENCES

/1/ B.W. Lampson, H.E. Sturgis, "Crash Recovery in a Distributed Storage System", unpublished paper, Computer Science Lab., Xerox Palo Alto Research Center, Palo Alto, CA, 1976.

/2/ W.H. Kohler, "A Survey of Techniques for Synchronization and Recovery in Decentralized Computer Systems", Computing Surveys, Vol. 13, No. 2, June 1981, pp. 149-183.

/3/ J.H. Wensley et al., "SIFT: Design and Analysis of a Fault-Tolerant Computer for Aircraft Control", Proceedings of the IEEE, Vol. 66, No. 10, Oct. 1978, pp. 1240-1255.

/4/ F. Farber, "Taskspecific Implementation of Fault Tolerance in Process Automation System", Proceedings of the Workshop on Self-Diagnosis and Fault-Tolerance, Tubingen, July 1981, pp. 84-102.

/5/ R.E. Glaser, G.M. Masson, "The Containment Set Approach to Crash-Proof Microprocessor Controller Design", Proceedings of the Twelfth International Symposium on Fault-Tolerant Computing, Santa Monica, CA, June 1982, pp. 215-222.

/6/ D.P. Siewiorek et al., "A Case Study of C.mmp, Cm* and C.vmp: Part I - Experiences with Fault Tolerance in Multiprocessor Systems", Proceedings of the IEEE, Vol. 66, No. 10, Oct. 1978, pp. 1178-1199.

/7/ F. Grandoni et al., "The MuTEAM System: General Guidelines", Proceedings of the Eleventh Annual International Symposium on Fault-Tolerant Computing, Portland, June 1981, pp. 15-16.

/8/ G. Cioffi, P. Corsini, G. Frosini, L. Lopriore, "MuTEAM: Architectural Insights of a Distributed Multimicroprocessor System", Proceedings of the Eleventh Annual International Symposium on Fault-Tolerant Computing, Portland, June 1981, pp. 17-19.

/9/ R.J. Swan, S.H. Fuller, D.P. Siewiorek, "Cm* - A Modular, Multimicroprocessor", Proceedings of the AFIPS 1977 National Computer Conference, AFIPS, Vol. 46, pp. 637-644.

/10/ M. Ajmone Marsan, G. Conte, D. Del Corso, F. Gregoretti, "Architecture, Communication Procedures and Performance Evaluation of the μ* Multimicroprocessor System", Proceedings of the First International Conference on Distributed Computing Systems, Huntsville, Alabama, 1979.

/11/ J. Archer Harris, D.R. Smith, "Hierarchical Multiprocessor Organizations", Proceedings of the Fourth Annual Symposium on Computer Architecture, 1977, pp. 41-48.

/12/ G. Mazarè, "MSC - A Symmetric Multi-Micro-Processor System", Proceedings of the Second Euromicro Symposium on Microprocessing and Microprogramming, Venice, 1976, pp. 135-140.

/13/ F. Baiardi, A. Fantechi, A. Tomasi, M. Vanneschi, "Mechanisms for a Robust Multiprocessing Environment in the MuTEAM Kernel", Proceedings of the Eleventh Annual International Symposium on Fault-Tolerant Computing, Portland, June 1981, pp. 20-24.

/14/ C.A.R. Hoare, "Communicating Sequential Processes", Communications of the ACM, Vol. 21, No. 8, Aug. 1978, pp. 668-679.

/15/ C.H. Sequin, A.M. Despain, D.A. Patterson, "Communication in X-TREE, A Modular Multiprocessor System", Proceedings of the ACM 1978 Annual Conference, Washington, D.C., Dec. 1978, pp. 194-203.

/16/ H. Sullivan, T.R. Bashkow, "A Large Scale, Homogeneous, Fully Distributed Parallel Machine", Proceedings of the Fourth Annual Symposium on Computer Architecture, 1977, pp. 105-117.

/17/ F. Baiardi, N. De Francesco, G. Vaglini, "A Remote Process Communication Facility for the MuTEAM System", ISI Technical Report, University of Pisa, 1982.

/18/ L. Simoncini, F. Sahebam, A.D. Friedman, "Design of Self--Diagnosable Multiprocessor Systems with Concurrent Computation and Diagnosis", IEEE Transactions on Computer, Vol. C-29, No. 6, June 1980, pp. 540-546.

/19/ P. Ciompi, F. Grandoni, L. Simoncini, "Distributed Diagnosis in Multiprocessor Systems: The MuTEAM Approach", Proceedings of the Eleventh Annual International Symposium on Fault-Tolerant Computing, Portland, June 1981, pp. 25-29.

/20/ B. Randell, P.A. Lee, P.C. Treleaven, "Reliability Issues in Computing System Design", Computing Surveys, Vol. 10, No. 2, June 1978, pp. 123-165.

/21/ T. Anderson, P.A. Lee, S.K. Shrivastava, "A Model for Recoverability in Multilevel Systems", IEEE Transactions on Software Engineering, Vol. SE-4, No. 6, Nov. 1978, pp. 486-493.

/22/ D.L. Russel, "State Restoration in Systems of Communicating Processes", IEEE Transactions on Software Engineering, Vol. SE-6, No. 2, Mar. 1980, pp. 183-194.

/23/ P. Corsini, L. Lopriore, L. Strigini, "Fail-Safeness in a Multiprocessor System: A Distributed Strategy Based on Backword Error Recovery", Technical Report, Italian National Computer Science Program, 1982.

REALIZING FAULT-TOLERANT BINARY TREES IN VLSI[1]

PETER J. VARMAN and DONALD S. FUSSELL
University of Texas at Austin
Austin, Texas 78712

ABSTRACT

The occurrence of production faults during the manufacturing process of an integrated circuit motivates the investigation of methods for configuring faulty VLSI processor arrays to obtain useful computational structures. In this paper, we study the algorithmic complexity of configuring a binary tree in a rectangular mesh of processors with arbitrary faults. The problem is shown to be NP-Complete.

INTRODUCTION

The binary tree has been recognized as a useful computational structure for solving complex "real world" problems [1] and several machines based on this structure have been designed viz. X-tree [2,3], P-Tree [4] and the Tree Machine [5]. A number of parallel algorithms for important problems like searching [6], sorting [5,7], determining the minimum spanning tree in a graph [8], graph coloring and matrix multiplication [1] etc. on a binary tree of processors have been discovered.

Advances in integrated circuit technology have raised the possibility of building wafer-sized systems of processing elements connected in the form of a binary tree. Some of these proposals have involved building a reconfigurable host network of processing elements connected in the form of a regular mesh, which are then programmed to be structured in any of several different configurations including the binary tree. The advantages of such an approach include low-cost VLSI implementation due to the regularity of the host network and its adaptability to several computational environments.

---

[1] This research was partially supported by NSF Grant Numbers MCS 81-09489 and MCS 81-04017

In either case the underlying problem is that of mapping a binary tree onto a two-dimensional surface and has been studied by several researchers [9-16]. Many of these studies were concerned with the problem of determining bounds on the area required by the two-dimensional layout of trees [9-13]. The mapping of the complete binary tree resulting in the so called H layout was shown in [6,7,13]. In [14], Horowitz and Zorat presented an algorithm for mapping an arbitrary binary tree onto the plane. The algorithm operates in time proportional to the number of nodes in the tree and may be used to configure a host network consisting of a rectangular grid of processors to obtain a specific binary tree. Koren [15] proposed building a rectangular grid of processing elements and described a distributed algorithm to structure a complete binary tree. Similarly, configuration of the CHIP computer, to obtain efficient binary tree interconnections was presented by Snyder [16].

The occurrence of production faults during the manufacturing process of an integrated circuit limits the maximum size of the chip that can be built with reasonable yield [1]. Reconfigurable processor arrays as proposed in [15-19], attempt to reduce the problem by exploiting the inherent redundancy of processors and data paths that are present in these structures. However, only a limited amount of systematic study on the problem of reconfiguring faulty arrays to obtain specific interconnection structures has been undertaken. Strategies for mapping linear arrays onto faulty rectangular (or hexagonal) mesh networks were presented in [15,17-19], for mapping grid networks in [18] and for complete binary trees in [15]. The solution in [15] proposed using each processor in the row and column in which a faulty processor occurred as a switching element to obtain an effective fault-free mesh of smaller dimensions. Since each faulty processor in an nxm rectangular mesh results in O(n+m) processors being made unavailable, the net yield (number of usable processors) using this approach would be small if more than a few processors were faulty.

This paper addresses the question of whether there exist efficient algorithms for embedding a given binary tree in a rectangular grid host network in which some subset of the processing elements may be faulty. We show here that this problem is NP-complete [20]. Determining efficient heuristics for specific types of fault patterns is the subject of ongoing research on fault tolerance in VLSI.

MODEL AND DEFINITIONS

The reconfigurable host network is modelled by an undirected rectangular mesh graph, with the vertices representing the processors and the edges the communication links between them. The network on which the binary tree is to be mapped is modelled by a connected component of the subgraph of the mesh, obtained by removing each vertex (and all the edges connected to it) that corresponds to a faulty processor. Configuring the network to obtain a given binary tree is equivalent to determining if there exists a planar embedding (homeomorphism) of the graph representing the binary tree in the given subgraph of the mesh.

Some definitions and results needed in the discussion are presented below.

An undirected graph G is a pair (V,E), where $V = \{v_1,v_2,\ldots,v_n\}$ is the set of *vertices* and $E \subseteq \{<v_i,v_j>, v_i \neq v_j\}$ is the set of *edges*. An edge $<v_i,v_j>$ is *incident* with the two vertices v_i and v_j. The *degree* of a vertex is the number of edges incident with it. A *path* in G is a sequence of edges $<v_{i_1},v_{i_2}>$, $<v_{i_2},v_{i_3}>$, ..., $<v_{i_{r-1}},v_{i_r}>$ for some $r \geq 2$. The *endpoints* of such a path are v_{i_1} and v_{i_r}. A path with endpoints v_{i_1} and v_{i_r} will be denoted by the ordered sequence of vertices $<<v_{i_1},v_{i_2},\ldots,v_{i_{r-1}},v_{i_r}>>$.

A *Rectangular Mesh*, M, is obtained by placing vertices at the Cartesian coordinates $\{(i,j) \,/\, 1 \leq i \leq P,\ 1 \leq j \leq Q\}$, with edges joining the pairs a unit distance apart.

A *Binary Tree*, T, is an acyclic graph in which exactly one vertex has degree two and the rest have degree either one or three.

A *Planar Embedding*[10] of $G = (V_G,E_G)$ in $H = (V_H,E_H)$ is a set of one-to-one mappings that map V_G into V_H, and E_G into paths in H, such that every such pair of paths is vertex disjoint except probably at the ends.

More formally, let $V_G = \{g_1,g_2,\ldots,g_n\}$, $V_H = \{h_1,h_2,\ldots,h_s\}$ and $P_H = \{ <<h_{i_1},h_{i_2}, \ldots, h_{i_{r-1}},h_{i_r}>> \,/\, h_{i_j} \varepsilon V_H\}$

Then, $F_1 : V_G \xrightarrow{1\text{-to-}1} V_H$, $F_2 : E_G \xrightarrow{1\text{-}1} P_H$ such that

1. $\forall <g_i,g_j> \varepsilon E_G$, $F_2<g_i,g_j> = <<F_1(g_i),h_{i_1},h_{i_2},\ldots,h_{i_r},F_1(g_j)>>$

2. $\forall \langle g_i,g_k\rangle, \langle g_j,g_l\rangle \varepsilon E_G$
if $F_2\langle g_i,g_k\rangle = \langle\langle F_1(g_i),h_{i_1},h_{i_2},\ldots,h_{i_r},F_1(g_k)\rangle\rangle$
and $F_2\langle g_j,g_l\rangle = \langle\langle F_1(g_j),h_{j_1},h_{j_2},\ldots,h_{j_m},F_1(g_l)\rangle\rangle$
then $h_{i_k} \neq h_{j_s} \neq F_1(g_t)$, $k=1,..,r$, $s=1,..,m$, $t=1,..,n$

Planar Cubic Hamiltonian Path Problem (3HP): Given a planar, cubic graph $G = (V,E)$, to determine if G has a Hamiltonian Path. This problem has been shown to be NP-complete [20].

Binary Tree Embedding Problem (BTE):
Given a Binary Tree $G = (V_G,E_G)$ and an arbitrary connected subgraph, $H = (V_H,E_H)$, of a rectangular mesh, to determine if G can be planar embedded in H.

RESULTS

We will show that BTE is NP-complete by reducing 3HP to it.

Let an arbitrary instance of 3HP be specified by a planar, cubic graph $G_0 =(V_0,E_0)$. Let $V_0 = \{v_1,v_2,\ldots,v_n\}$ where $\text{degree}(v_i) = 3$. It is required to determine if G_0 has a Hamiltonian Path.

Construct an instance of the BTE problem as follows.

(1) Planar embed G_0 in a rectangular mesh using, for instance, Valiant's [10] construction. Remove all vertices and edges from the mesh that are not part of the embedding of G_0. Let the resulting subgraph of the mesh be $H = (V_H,E_H)$ (Fig.1). This construction can be done in polynomial time.

The vertices V_H of H are of two distinct types - those which are images of the vertices in V_0 and those which lie on paths between these vertices. Let the first set be denoted by $A = \{a_1,a_2,\ldots,a_n\}$ where a_i is the image of v_i, and the second by the set B. Note that $\text{degree}(a_i) = 3$, for all $a_i \varepsilon A$ and the degree of each vertex in B is 2. Further with each vertex a_i in A associate the set $S_i = \{b_i^+,b_i^-,b_i^*\}$ such that $\langle a_i,b_i^+\rangle$, $\langle a_i,b_i^-\rangle$ and $\langle a_i,b_i^*\rangle \varepsilon E_H$. Note that by the construction $S_i \cap S_j = \Phi$, for $i \neq j$, and the number of vertices from B in the path between a_i and a_j (which is the image of edge $\langle v_i,v_j\rangle$ in E_0) is greater than three.

Let $P(a_i,a_j)$ denote a path between a_i and a_j in H containing intermediate vertices only from the set B i.e.
$P(a_i,a_j) = \langle\langle a_i,b_{i_1},b_{i_2},\ldots,b_{i_r},a_j\rangle\rangle$, $r \geq 3$, $b_{i_k} \varepsilon B$. The embedding of G_0 in H possesses the following property.

Property 1:
Between any two vertices $a_i,a_j \varepsilon A$, there exists at most one path of the form $P(a_i,a_j)$. This path exists if and only if $\langle v_i,v_j\rangle \varepsilon E_0$.

(2) Construct a Binary Tree $G = (V_G,E_G)$ as follows.
$V_G = \{r_0,d_0,g_1,g_2,\ldots,g_n, d_1,d_2,\ldots,d_{n+1}\}$
$E_G = \{\langle r_0,d_0\rangle, \langle r_0,g_1\rangle, \langle g_i,g_{i+1}\rangle$ $i=1,\ldots,n-1$, $\langle g_i,d_i\rangle$ $i=1,\ldots,n$, $\langle g_n,d_{n+1}\rangle\}$.

Lemma 1

G_0 has a Hamiltonian Path if and only if G is planar embeddable in H.

Proof

(IF-Part)
Suuppose G is planar embedded in H (Fig. 2). Since the only vertices in H of degree greater than two are $a_1,\ldots,a_n$ and each $g_i \varepsilon E_G$ has degree three, the mapping F_1 must map each of these vertices onto a distinct vertex from the set A.
i.e $F_1(g_i) = a_{j_i}$, $i=1,\ldots,n$
The edge $\langle g_i,g_{i+1}\rangle$ is mapped onto a path in H between a_{j_i} and $a_{j_{i+1}}$. This path cannot contain any vertex belonging to A as an intermediate vertex, since that vertex would then lie on two paths in the embedding, contradicting the planarity requirement. Thus, the edge $\langle g_i,g_{i+1}\rangle$ must be mapped onto a path $P(a_{j_i},a_{j_{i+1}})$, and by Property 1 this implies that $\langle v_{j_i},v_{j_{i+1}}\rangle \varepsilon E_0$. Since $\langle g_i,g_{i+1}\rangle \varepsilon E_G$ for $i=1,\ldots,n-1$, $\langle v_{j_i},v_{j_{i+1}}\rangle$ $i=1,\ldots,n-1$ are edges in G_0.
Therefore $\langle\langle v_{j_1},v_{j_2},\ldots,v_{j_n}\rangle\rangle$ is a Hamiltonian Path.

(ONLY-IF Part)
Suppose G_0 has a Hamiltonian Path $\langle\langle v_{j_1},v_{j_2},\ldots,v_{j_n}\rangle\rangle$, $v_{j_i} \varepsilon V_0$.
We construct a planar embedding of G in H as follows (Fig.2).

By Property 1, for all $i=1,\ldots,n-1$, $P(a_{j_i},a_{j_{i+1}})$ is a path in H that has only intermediate vertices from B. Furthermore, since G_0 was planar embedded in H, no intermediate vertex lies on any two paths. Hence the mapping

$F_1(g_i) = a_{j_i} \quad i=1,\dots,n$

$F_2\langle g_i,g_{i+1}\rangle = P(a_{j_i},a_{j_{i+1}}) \quad i=1,\dots,n-1$

is a planar embedding.

Now for each of the vertices a_{j_i}, $i=2,\dots,n-1$ there exists one vertex in S_{j_i} that has not been mapped as above. Let this be $b_{j_i}^{+}$. Also each of the vertices a_{j_1}, a_{j_n} have two vertices belonging to S_{j_1} and S_{j_n} respectively that have not been so mapped. Let these be $b_{j_1}^{+}$, $b_{j_1}^{*}$ and $b_{j_n}^{+}$, $b_{j_n}^{*}$ respectively.

Define the mappings

$F_1(d_i) = b_{j_i}^{+} \quad i=1,\dots,n$

$F_2\langle g_i,d_i\rangle = \langle a_{j_i},b_{j_i}^{+}\rangle \quad i=1,\dots,n$

$F_1(d_{n+1}) = b_{j_n}^{*}$

$F_2\langle g_n,d_{n+1}\rangle = \langle a_{j_n},b_{j_n}^{*}\rangle$

$F_1(r_0) = b_{j_1}^{*}$

$F_2\langle g_1,r_0\rangle = \langle a_{j_1},b_{j_1}^{*}\rangle$

Finally, since each path $P(a_i,a_j)$ in H contains at least three vertices from B, there is a vertex $b^{**} \varepsilon B$ such that $\langle b_{j_1}^{*}, b^{**}\rangle \varepsilon E_H$ and b^{**} has not been mapped.

Let $F_1(d_0) = b^{**}$

$F_2\langle d_0,r_0\rangle = \langle b_{j_1}^{*}, b^{**}\rangle$

Clearly the set of mappings described above form a planar embedding of G in H.

Theorem: BTE is NP-complete.

Proof:

BTE is in NP since a nondeterministic algorithm need only guess a mapping of vertices and paths for the embedding and check in polynomial time that this is a valid planar embedding.

Then by the polynomial reduction of any instance of 3HP to BTE as described above and Lemma 1, the theorem follows.

CONCLUSIONS

Reconfigurable Processor Arrays are an effective method of utilizing the potential for highly parallel processing made possible by VLSI technology. However, due to the exponential increase in the probability of a fatal production flaw with the size of the circuit, it is desirable that strategies for configuring useful computational structures in the presence of faulty elements be investigated. In this paper we have studied the algorithmic complexity of structuring an important interconnection structure -the binary tree- in an underlying rectangular mesh network with arbitrary faults. The problem has been shown to be NP-complete. Further work to obtain efficient heuristics and/or architectural strategies is needed to fully exploit the potential of reconfigurable architectures.

REFERENCES

1. C. Mead and L. Conway, "Introduction to VLSI Systems," Reading, MA, Addison-Wesley 1980.

2. A.Despain and D. Patterson, "X-tree: A structured multiprocessor computer architecture," Proc. 5th Annual Symp. on Computer Architecture, Apr. 1978.

3. C. Sequin, "Single Chip Computers, the new VLSI Building Blocks," Proc. Caltech. Conf. on VLSI, Jan. 1979.

4. J. Harris and D. Smith, "Simulation experiments of a tree organized multicomputer," Proc. 6th Annual Symp. on Computer Architecture, Apr. 1979.

5. S. Browning, "The Tree Machine: A Highly Concurrent Programming Environment," PhD Thesis, Caltech, Jan. 1980.

6. J.L. Bentley and H.T. Kung, "A Tree Machine for Searching Problems," Proc. 1979 Int'l Conf. Parallel Processing, Aug. 1979.

7. C.E. Leiserson, "Systolic Priority Queues," Proc. Caltech Conf. on VLSI, Jan. 1979.

8. J.L. Bentley, "A Parallel Algorithm for Constructing Minimum Spanning Trees" Journal of Algorithms, Jan. 1980.

9. R.P. Brent and H.T. Kung, "On the Area of Binary Tree Layouts," Information Processing Letters, Vol. 11, No. 1, Aug. 1980.

10. L.G. Valiant, "Universality considerations in VLSI Circuits," IEEE Trans. on Computers, Vol. C-30, No. 2, Feb. 1981.

11. M.S. Paterson, W.L. Ruzzo and L. Snyder, "Bounds on Minimax

Edge Length for Complete Binary Trees," *Proc. 13th Annual Symp. on the Theory of Computing, 1981.*

12. W.L. Ruzzo and L. Snyder, "Minimum Edge Length Planar Embeddings of Trees,"*VLSI Systems and Computations, H.T.Kung, R.F.Sproull, G.L.Steele Jr. eds., CMU, Computer Science Press, Oct. 1981*

13. M. Rem, "Mathematical Aspects of VLSI Design," *Proc. Caltech Conf. on VLSI, Jan. 1979.*

14. E. Horowitz and A. Zorat, "The Binary Tree as an Interconnection Network: Applications to Multiprocessor Systems and VLSI," *IEEE Trans. on Computers, Vol. C-30, No. 2, Apr. 1981.*

15. I. Koren, "A Reconfigurable and Fault-Tolerant VLSI Multiprocessor Array," *Proc. 8th Annual Symp. on Computer Architecture, Apr. 1981.*

16. L. Snyder, "Introduction to the Configurable, Highly Parallel Computer," *Computer, Vol. 15, No. 1, Jan. 1982.*

17. D. Fussell and P. Varman, "Fault-Tolerant Wafer-Scale Architectures for VLSI," *Proc. 9th Annual Symp. on Computer Architecture, Apr. 1982.*

18. F.B. Manning, "An Approach to Highly Integrated, Computer-Maintained Cellular Arrays," *IEEE Trans. on Computers, Vol. C-26, No. 6, June 1977.*

19. R.C. Aubusson and I. Catt, "Wafer-Scale Integration - a Fault-Tolerant Procedure," *IEEE J. Solid-State Circuits, Vol. SC-13, No. 3, June 1978.*

20. M.R. Garey and D.S. Johnson, "Computers and Intractability - A Guide to the Theory of NP-Completeness," *W.H.Freeman and Company,1979.*

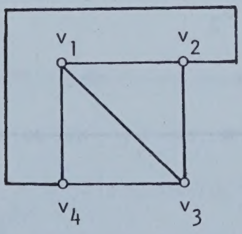


Fig. 1(a): A planar cubic graph.

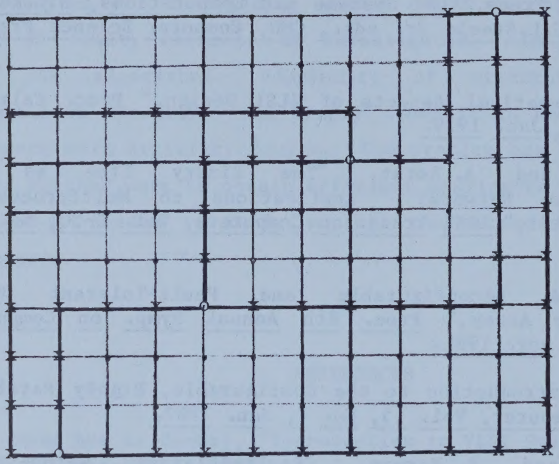


Fig. 1(b): Embedding of Graph of Fig. 1(a) in a mesh.

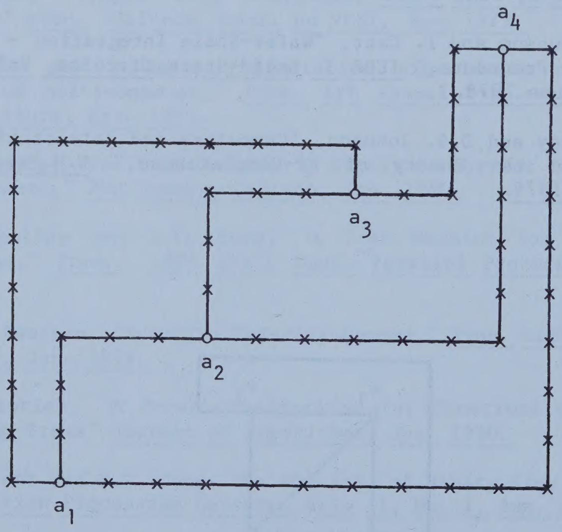


Fig. 1(c): Mesh subgraph H. (Nodes denoted by x belong to set B)

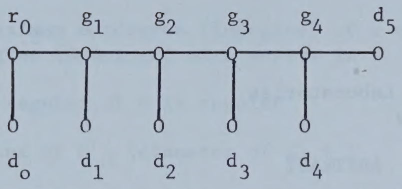

r0
g1
g2
g3
g4
d5
do
d1
d2
d3
d4

Fig. 2(a): Binary Tree G to be embedded in H of Fig. 1(c).

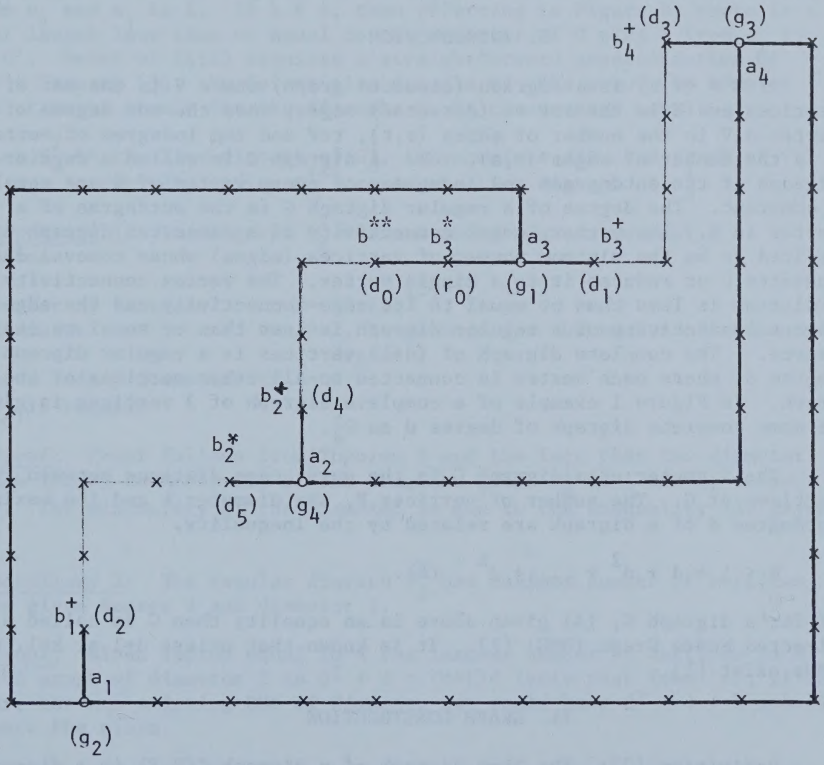

b4+
(d3)
(g3)
a4
b**
b3*
a3
b3+
(d0)
(r0)
(g1)
(d1)
b2+
(d4)
b2*
a2
(d5)
(g4)
b1+
(d2)
a1
(g2)

Fig. 2(b): Embedding of G in H corresponding to the Hamiltonian Path (v_3, v_1, v_4, v_2).

(Nodes in parenthesis correspond to the mapped nodes of G)

ON DIGRAPHS WITH MINIMUM DIAMETER AND MAXIMUM CONNECTIVITY

S.M. REDDY, J.G. KUHL AND S.H. HOSSEINI*
Electrical and Computer Engineering
University of Iowa
Iowa City, Iowa

and

H. LEE, Bell Telephone Laboratories
Murray Hill, New Jersey

ABSTRACT

It is shown that the line digraphs of the complete digraphs give digraphs that are denser than the currently known digraphs of the same degree and diameter. We also show that the connectivity of these digraphs is maximum. We introduce the notion of t-deleted diameter and establish upper bounds on the deleted diameter of the newly constructed digraphs.

I. INTRODUCTION

If $G = (V,E)$ is a digraph (directed graph) where V is the set of vertices and E is the set of (directed) edges, then the out degree of a vertex $s \varepsilon V$ is the number of edges (s,t), $t \varepsilon V$ and the indegree of vertex s is the number of edges (t,s), $t \varepsilon V$. A digraph G is called a regular digraph if the outdegrees and indegrees of every vertex of G are equal to a constant. The degree of a regular digraph G is the outdegree of a vertex in G. The vertex (edge) connectivity of a connected digraph G is defined to be the minimum number of vertices (edges) whose removal disconnects G or reduces it to a single vertex. The vertex connectivity of a digraph is less than or equal to its edge connectivity and the edge and vertex connectivity of a regular digraph is less than or equal to its degree. The complete digraph of (d+1) vertices is a regular digraph of degree d, where each vertex is connected to all other vertices of the graph. In Figure 1 example of a complete digraph of 3 vertices is given. We name complete digraph of degree d as G_d^1.

The diameter of a digraph G is the worst case distance between two vertices of G. The number of vertices N, the diameter k and the maximum outdegree d of a digraph are related by the inequality.

$$N \leq 1 + d + d^2 + \ldots + d^k \quad (A).$$

If for a digraph G, (A) given above is an equality then G is called a Directed Moore Graph (DMG) [1]. It is known that unless d=1 or k=1, no DMGs exist [1].

II. GRAPH CONSTRUCTION

Definition [2]: The line digraph of a digraph G(V,E) is a digraph $G'(V',E')$ where $V' = E$ and $E' = \{(s,t): s = (q_a,q_b) \varepsilon E$ and $t = (q_b,q_c) \varepsilon E\}$.

Note that the edges of the line digraphs of a digraph G are paths of length two in G. The line digraph of the complete digraph of Figure 1 is given in Figure 2.

---

*The research by SMR, JGK and SHH was supported by AFOSR Grant AFOSR-78-3582 and NSF Grant ECS-8205188.

Theorem 1 given below relates the parameters of a line digraph of a digraph G to the parameters of G.

Theorem 1: Let $G' = (V',E')$ be the line digraph of a digraph G. Then

(i) the maximum outdegree (indegree) of a vertex in $V' \leq$ maximum outdegree (indegree) of a vertex in V

(ii) G' is regular if G is regular

(iii) diameter of $G' \leq$ (diameter of G) + 1

(iv) node (edge) connectivity of $G' \geq$ node (edge) connectivity of G.

Proof: Proofs of (i) and (ii) are straightforward. To prove (iii) let $s_i, s_j, \varepsilon V', s_i \neq s_j$. Then there exists two edges in E, say (a,b) and (c,d), such that $(a,b) = s_i$ and $(c,d) = s_j$. If $b = c$ then the distance between s_i and s_j is 1. If $b \neq c$, then referring to Figure 3, there is a path of length less than or equal to the diameter of G plus 1 from s_i to s_j in G'. Proof of (iii) requires a straightforward generalization of Menger's Theorem [3]. We do not give details of this proof (we however give a stronger result for the digraphs to be constructed next).

Let G_d^2 be the line digraph of G_d^1, the complete directed graph of degree d. Iteratively define G_d^{i+1} to be the line digraph of G_d^i.

Corollary 1: The digraph G_d^i defined above has the following properties.

number of vertices = $(d+1)d^{i-1}$
diameter = i, is minimum for the number of nodes and degree
node (edge) connectivity = d
G_d^i is regular

Proof: Proof follows from Theorem 1 and the fact that the diameter and node (edge) connectivity of the complete graph G_d^1 is 1 and d, respectively. The minimality of the diameter is due to the inequality (A) given above.

Corollary 2: The regular digraph G_d^2 has maximum number of vertices for the given degree d and diameter 2.

Proof: Given degree equal to d the largest number of vertices in a directed graph of diameter 2 is $d^2 + d = (d+1)d$ (note that from (A), given earlier, the non-existing DMG of diameter two would have $d^2 + d + 1$ nodes) and hence the claim.

It is not known whether the G_d^i graphs are the densest (i.e. have the largest number of vertices for given degree and diameter) for diameters > 2.

III. PATHS IN G_d^i

We next want to establish that between every pair of nodes in G_d^i there are d vertex disjoint paths of maximum length (i+2).

Theorem 2: Between every pair of nodes in G_d^i there are at least d vertex disjoint paths, of which at least (d-1) paths are of length less than or equal to (i+1) and one path of length less than or equal to (i+2).

Proof: Proof is by finite induction on i, the iteration index (and the diameter) of G_d^i. Clearly the theorem is valid for G_d^1. Assume the theorem is true for all G_d^i, $i \leq k - 1$. Let s and t be two vertices in G_d^k. Then there are two edges, say (q_1, q_2) and (q_3, q_4) in G_d^{k-1}, such that $(q_1, q_2) = s$ and $(q_3, q_4) = t$.

Case 1: Let $q_2 \neq q_3$. Then there are d vertex disjoint paths, from q_2 to q_3, in G_d^{k-1} with the following properties.:

(i) at least (d-1) paths have length $\leq k$

(ii) one path of length $\leq k + 1$ and

(iii) neither the edge (q_1, q_2) nor the edge (q_3, q_4) occurs on these paths.

As shown in Figure 4, d vertex disjoint paths of desired length, can be identified from vertex s to vertex t in G_d^k from the paths in G_d^{k-1} cited above.

Case 2: $q_2 = q_3$. In this case t is an immediate successor of s. Hence, we only need to exhibit (d-1) other vertex disjoint paths among which at least (d-2) paths have length $\leq k + 1$ and one has length $\leq k + 2$. Assume this is not the case. Then (referring to Figure 5(a)) there do not exist (d-1) edge disjoint cycles, including q_2, in G_d^{k-1} simultaneously satisfying the following conditions:

$$(B)\begin{cases} \text{(i) none of the cycles include s or t and} \\ \text{(ii) (d-2) cycles are of length} \leq k \text{ and 1 cycle is of length} \leq k + 1. \end{cases}$$

This would imply that in G_d^{k-2}, referring to Figure 5(b), there do not exist (d-1) edge disjoint walks from v_3 to v_2 that possess the following properties, simultaneously:

(i) none of the walks start with q_4 or end with q_1

(ii) none contain the subpaths $q_2 \rightarrow q_4$ or $q_1 \rightarrow q_2$ and

(iii) d-2 are of length $\leq$ k-1 and one is of length k.

By working iteratively backwards, the conditions given in (B) above, would imply that in G_d^1 referring to Figure 5(c), there do not exist (d-1) edge disjoint walks from r_k to r_2, which simultaneously possess the following properties:

$$(C)\begin{cases} \text{(i) none of the walks start with } p_k \text{ nor end with } p_1 \\ \text{(ii) none of the walks contain } p_1 \to p_2 \to \dots \to p_{k-1} \text{ nor } p_2 \to p_3 \to \dots \\ \qquad \to p_k \text{ as subpaths and} \\ \text{(iii) } d-2 \text{ walks are of length } \leq 2 \text{ and } 1 \text{ is of length } \leq 3. \end{cases}$$

It can be shown that walks admitting condition (C) given above do exist in G_d^1 and hence the theorem is proved.

IV. REMARKS

We have constructed a class of dense diagraphs with minimum diameter and maximum connectivity. The G_d^i graphs have $[1 + 1/d]$ times as many vertices as the densest known digraphs of the same diameter (the earlier known digraphs are the de Bruijn graphs of d^i vertices [4]). Furthermore these new digraphs have maximum connectivity.

Another interesting property of the G_d^i graphs is that if any arbitrary combination of (d-2) vertices and/or edges of the graphs are removed, the diameter of the resulting graph is at most one greater than the diameter of the original graph and if an additional edge or vertex is removed the diameter increases by atmost 1. This property was formally stated in Theorem 2. The diameter of the graphs resulting from the deletion of vertices and/or edges has been earlier called t-deleted diameter [5] and has applications in fault-tolerant distributed computing networks and communication networks.

The de Bruijn graphs are known to represent the state diagrams of shift registers with alphabet size of d. The G_d^i graphs can be portrayed as the state diagrams of the shift registers shown in Figure 6. The labels of vertices would be restricted sequences of length i over an alphabet of size (d+1). A sequence $\langle a_o, a_1, a_2, \dots, a_{i-1} \rangle$ over an alphabet of size (d+1) is a label for a vertex in the G_d^i graph if and only if $a_{r+1} = a_r + j$ modulo $(d+1)$, $1 \leq j \leq d$ and $0 \leq r \leq i - 2$. The set of vertex labels for $d = 2$ and $i = 3$ are given in Figure 7. Note that the number of usch restricted sequences of length i is $(d+1)d^{i-1}$. The immediate successors of vertices in G_2^3 are also given in Figure 7.

Yet another characterization of the G_d^i graphs is that they are the state diagrams of the sequential circuits given in Figure 8. The vertex labels and the labels for the G_2^4 graph in this representation is given in Figure 9.

The last two representations of G_d^i graphs immediately indicate the diameter of these graphs to be i. However neither of these representations has proved to be helpful in deriving the results on connectivity.

REFERENCES

1. S. Toueg, "On Directed Moore Graphs," Department of Electrical Engineering, Princeton University, Princeton, N.J.

2. F. Harary and R.Z. Norman, "Some Properties of Line Digraphs," Rend. Circ. Mat. Palermo, 9 (1961), pp. 161-168.

3. Claude Berge, "Graphs and Hypergraphs," North-Holland Publishing Company, 1973.

4. N. Deo, "Graph Theory with Applications to Engineering and Computer Science," Prentice Hall, Inc., 1974.

5. J. Kuhl, "Fault Diagnosis in Computing Networks," Ph.D. Thesis, Department of Computer Science, University of Iowa, Iowa City, Iowa, August 1980.

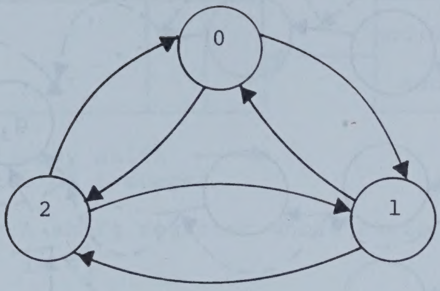


Figure 1: A 3 node complete directed graph

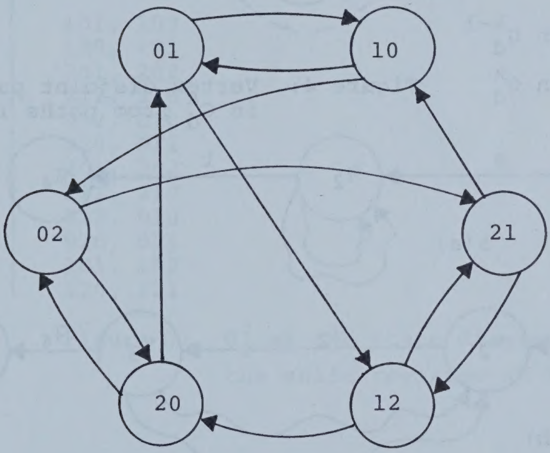

01
10
02
21
20
12

Figure 2: Line digraph of the complete directed graph of 3 nodes

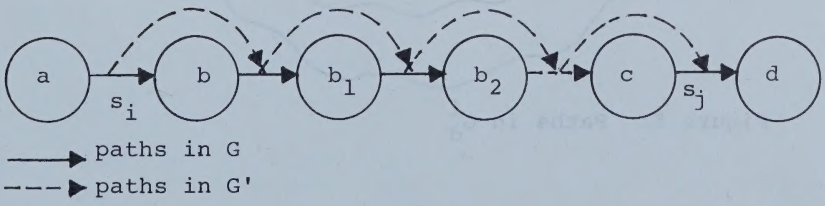

a
b
b1
b2
c
d
si
sj
paths in G
paths in G'

Figure 3: Paths in G and G'

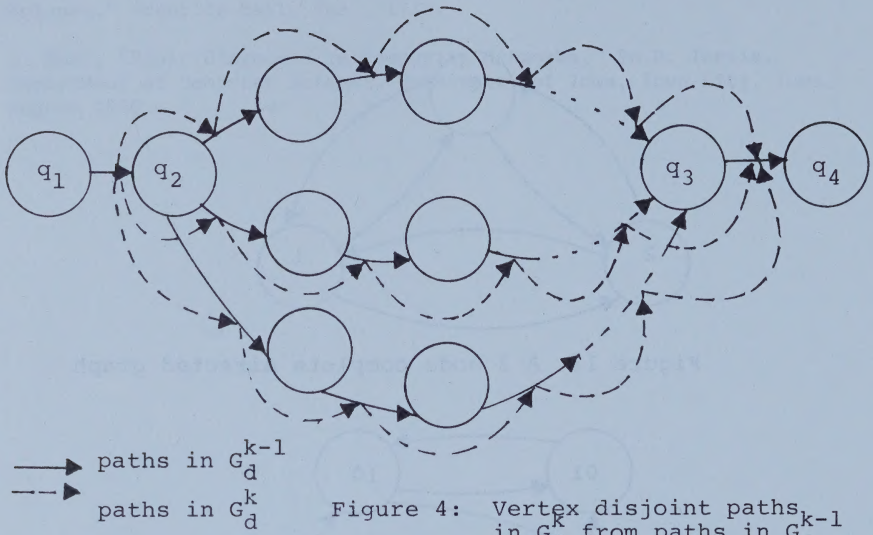

q_1
q_2
q_3
q_4
paths in G_d^{k-1}
paths in G_d^k

Figure 4: Vertex disjoint paths in G_d^k from paths in G_d^{k-1}

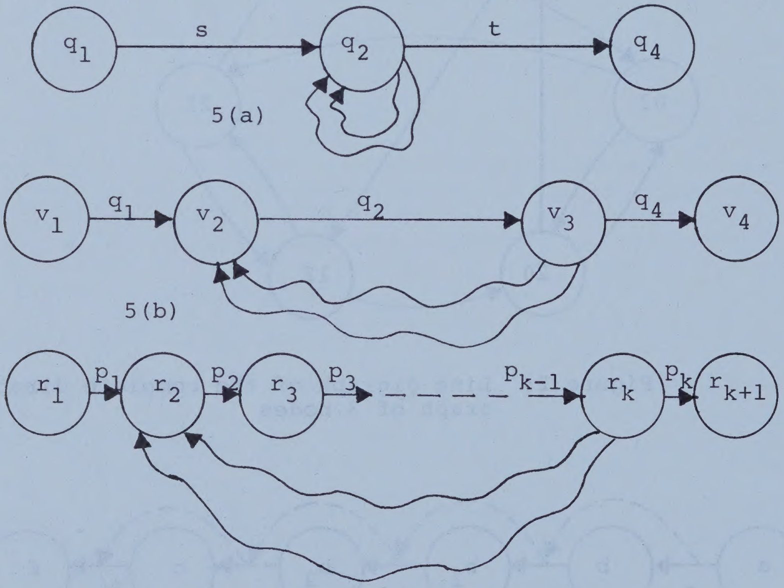

q_1
s
q_2
t
q_4
5(a)
v_1
q_1
v_2
q_2
v_3
q_4
v_4
5(b)
r_1
p_1
r_2
p_2
r_3
p_3
p_{k-1}
r_k
p_k
r_{k+1}

Figure 5: Paths in G_d^i

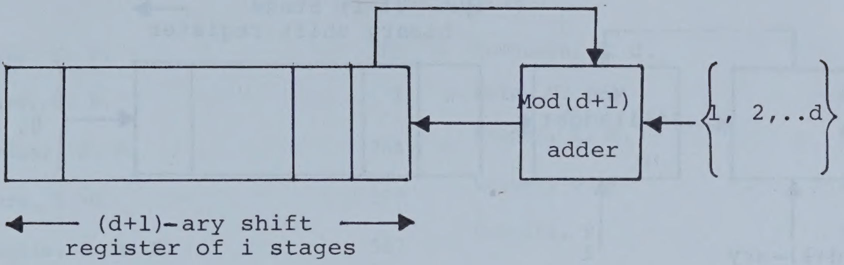

Mod(d+1)
adder
1, 2,..d
(d+1)-ary shift
register of i stages

Figure 6: A shift register whose state diagram in G_d^i

Node	Successors
010	101, 102
012	120, 121
020	201, 202
021	212, 210
101	012, 010
102	020, 021
120	201, 202
121	212, 210
201	012, 010
202	020, 021
210	101, 102
212	120, 121

Figure 7: G_2^3 as the state diagram of the shift register in Figure 6.

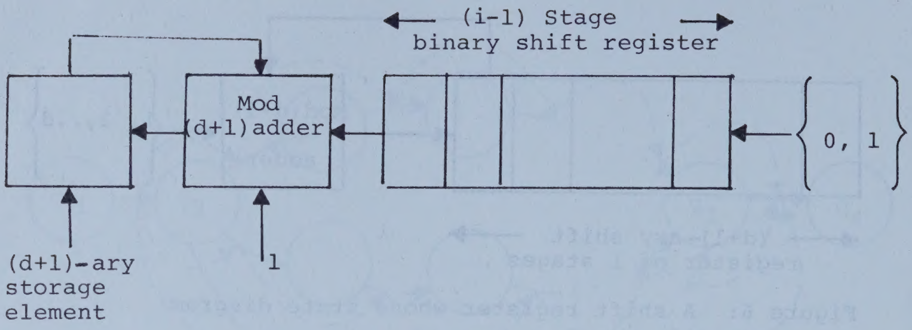

(i-1) Stage
binary shift register
Mod
(d+1) adder
0, 1
(d+1)-ary
storage
element
1

Figure 8: Another sequential circuit whose state diagram is G_d^i

Node	Successors
0000	1000, 1001
0001	1010, 1011
0010	1100, 1101
0011	1110, 1111
0100	2000, 2001
0101	2010, 2011
0110	2100, 2101
0111	2110, 2111
1000	2000, 2001
1001	2010, 2011
1010	2100, 2101
1011	2110, 2111
1100	0000, 3001
1101	0010, 3011
1110	0100, 3101
1111	0110, 3111
2000	0000, 3001
2001	0010, 3011
2010	0100, 3101
2011	0110, 3111
2100	1000, 1001
2101	1010, 1011
2110	1100, 1101
2111	1110, 1111

Figure 9: G_2^4 as the state diagram of the sequential circiut of the shift register in Figure 8

INDEX OF AUTHORS

Abaya, E. F. 281
Agnew, G. B. 1
Agrawal, D. P. 704
Akers, S. G. 219
Alegria, J. A. S. 567
Alexandris, N. 5
Ammar, G. 187
Antsaklis, P. J. 341
Apostolico, A. 924
Bayoumi, M. 150
Bentley, J. L. 452
Bingulac, S. 159
Blank, L. 967
Brown, C. A. 943
Brown, M. E. 567
Bruzzone, S. 555
Buzacott, J. A. 444
Cao, X. 434
Carlson, D. A. 374
Carter, M. J. 680
Chandra, S. 209
Chazelle, B. M. 615
Chen, M. S. 27
Cheng, J.-S. 965
Chu, C.-H. H. 116
Chung, M.-J. 888
Cobb, J. D. 825
Cohen, D. 126
Colbourn, C. J. 25
Cole, C. 452
Cooper, E. H. 955
Cooper, G. R. 955
Corsini, P. 998
Culik, K. 599
Czarnecki, S. V. 759
Dalili-Yazdi, A. 906
da Rocha, J. R. F. 985
Davies, O. J. 987
Dehne, F. 85
De Lorenzo, M. 803
Delp, E. J. 116, 135
Dhall, S. K. 906
Dickinson, B. W. 524, 874
Dimitriadis, B. 5
Dunham, J. G. 688
Duarte, M. 967
Eichel, P. H. 135
Farvardin, N. 290
Fiduccia, C. M. 272, 934
Frederickson, G. N. 490
Friesen, D. K. 941
Fussell, D. S. 625, 1008
Georghiades, C. N. 975
Giancarlo, R. 924
Gibson, J. D. 301
Gonzalez, T. F. 636
Goodman, R. M. F. 672

Hakimi, S. L. 231
Halverson, D. R. 733
Hibey, J. L. 345
Hirschberg, D. S. 257
Hosseini, S. H. 1018
Huang, S.-H. S. 481
Huang, Y.-F. 769
Hwang, I. S. 241
Jin, X. 627
Johnson, D. B. 898
Johnsson, L. 126
Jounnaud, H.-P. 396
Joyce, L. S. 97
Juang, J. N. 321
Jullien, G. A. 150
Kamel, M. 884
Kaminski, T. J. 577
Kasami, T. 406
Kashyap, R. L. 355
Kassam, S. A. 526,945
Kaveh, M. 555
Kennedy, M. A. 206
Keutzer, K. 264
Khalil, H. K. 843
Kime, C. R. 241
Kirchner, C. 396
Kirchner, H. 396
Kirkpatrick, D. 35
Komandur, S. M. 704
Krieger, M. 793
Krishna, C. M. 996
Krishnamoorthy, M. S. 888
Krishnamurthy, B. 219
Kuhl, J. G. 1018
Kundu, S. 384
Kwong, Y.-S. 472
Lakani, G. D. 607
Langston, M. A. 941
Lasdon, L. S. 416
Laskin, R. A. 813
Lazar, A. A. 3,15
Lee, H. 1018
Lee, S. L. 636
Leighton, T. 500
Leon-Garcia, A. 977
Lepley, M. 500
Likins, W. 813
Lindahl, C. E. 343
Lingas, A. 53
Litkouhi, B. 843
Lloyd, E. L. 646
Lombardi, F. 987
Longman, R. W. 321,331,779,813
Loomis, H. H., Jr. 557
Lopiore, L. 998
Luccio, F. 914
Mark, J. W. 1
Martin, C. 187

Martinez, A. 749
Mathews, V. J. 545
Mattheyses, R. M. 272
Mazor, B. 689
Mears, B. C. 823
Medanic, J. 159
Meditch, J. S. 27
Meyer, G. 567
Meyer, G. G. L. 206
Mintz, M. 516
Modestino, J. W. 290
Mohan, S. 704
Moustakides, G. V. 713,945
McEliece, R. J. 672
McGillem, C. D. 107
Nakajima, K. 231,656
Natour, I. A. 599
Neuhoff, D. L. 680
Normura, T. 177
Olsder, G. L. 788
O'Reilly, J. J. 967,985
O'Rourke, J. 75
Pagli, L. 914
Papavassilopoulos, G. P. 788
Pearlman, W. A. 689
Petersen, I. R. 196
Pinter, R. Y. 53
Pless, V. 682
Pollara, R. 507
Pollock, G. G. 557
Polyzos, G. 5
Pomalaza, C. A. 107
Pupolin, S. 957
Purdom, P. 943
Quatember, B. 589
Rahnamai, K. 791
Ramakrishnan, I. V. 625
Rappaport, D. 43
Ravi, S. S. 646
Reddy, S. M. 1018
Reid, R. E. 823
Rivest, R. 53
Robertazzi, T. G. 535
Robertson, E. 264
Root, W. L. 97
Ross, A. A. 557
Saberi, A. 855
Sack, J.-R. 64
Sain, M. K. 153
Saluja, K. K. 209,229
Santoro, N. 251,793
Savage, J. E. 665
Sawan, M. E. 791,886
Sayood, K. 301
Schwartz, S. C. 535
Seidel, R. 35,94
Sennott, L. I. 365
Shamir, A. 53

Shin, K. G. 996
Sid-Ahmed, M. A. 150
Sidney, J. G. 251
Silberschatz, A. 625
Simaan, M. 144
Singhal, K. 884
Sira-Ramirez, H. 167,339
Sirlin, S. W. 321
Skelton, R. E. 803,865
Smith, J. E. 577
Snyder, D. L. 965,975
Stark, W. E. 697
Stecke, K. E. 426
Strigini, L. 998
Sun, M. 656
Suri, R. 434
Suzuki, I. 406
Swaszek, P. F. 311
Tantaratana, S. 723
Tarn, T. J. 834
Thomas, J. B. 713,759
Tomasi, C. 957
Tran, M. T. 886
Tugcu, A. K. 823
Tugnait, J. K. 545
Vaccaro, R. J. 874
Vakil, F. 15
Vannicola, V. C. 363
Varman, P. J. 1008
Varshney, P. K. 363
Vastola, K. S. 739
Veloso, P. A. S. 394
Venkatesan, S. M. 898
Venkatraman, C. S. 229
Wald, J. 25
Waren, A. D. 416
Warwick, K. 341
Weide, B. 567
Willard, D. 462
Wise, G. L. 281,366
Wong, C. K. 481
Wood, D. 472
Wyman, B. F. 153
Yan, G. 834
Yao, D. W. 444
Yao, D. D. W. 977
Yao, K. 507
Yeh, P. C. C. 587
Yousuff, A. 865
Zeytinoglu, M. 516
Zhang, Y. G. 331
Zhu, Z. T. 779

UNIVERSITY OF ILLINOIS-URBANA
621.3811AL5 C001
PROCEEDINGS / ALLERTON CONFERENCE ON COM
1982
3 0112 008149178